Τμήμα Πληροφορικής και Τηλεματικής
Χαροκόπειο Πανεπιστήμιο
ΕΠ34 Εφαρμογές Τεχνητής Νοημοσύνης

Απαλλακτική Εργασία Σεπτεμβρίου 2021

Έκδοση 2021-1.0

Διδάσκων: Χρήστος Δίου

Σχετικά με την εργασία

Η εργασία αποτελείται από δύο ανεξάρτητα θέματα που έχουν ως στόχο να αξιολογήσουν τις γνώσεις σας σε όσα διδαχθήκατε στο μάθημα. Για κάθε ερώτημα καλείστε να υποβάλλετε ένα από τα παρακάτω στο eclass (υποβολή στην ενότητα "Εργασίες" του μαθήματος).

1. Ένα python notebook για κάθε ερώτημα (σύνδεσμος στο colab ή μέσω αρχείου ipynb) το οποίο να περιέχει τον κώδικα και τις απαντήσεις στις ερωτήσεις ή, εναλλακτικά
2. Ένα αρχείο python για κάθε ερώτημα με τον κώδικα, και ένα αρχείο PDF με απαντήσεις στις ερωτήσεις

Σε περίπτωση που δεν έχετε πρόσβαση σε υπολογιστή με γρήγορο επεξεργαστή ή/και επιτάχυνση GPU σας συστήνεται να χρησιμοποιήσετε το Google Colab για την ανάπτυξη της εργασίας. Μπορείτε να χρησιμοποιήσετε Runtime Environment με κάποιον accelerator (GPU ή TPU).

1 Προεπεξεργασία δεδομένων και multi-layer perceptron

Θεωρήστε το σύνολο δεδομένων "Census Income", η περιγραφή του οποίου βρίσκεται στον παρακάτω σύνδεσμο:

https://archive.ics.uci.edu/ml/datasets/Census+Income

Στόχος μας είναι να κατασκευάσουμε ένα μοντέλο που προβλέπει αν κάποιος κάτοικος των Η.Π.Α. έχει ετήσιο εισόδημα μεγαλύτερο από 50.000 δολάρια αξιοποιώντας δημογραφικά χαρακτηριστικά, με βάση απογραφικά δεδομένα του 1994.

1.1 Φόρτωση δεδομένων

Αρχικά, παρατηρούμε ότι το σύνολο δεδομένων αποτελείται από έναν συνδυασμό αριθμητικών και κατηγορικών μεταβλητών. Γράψτε μία συνάρτηση

```
X_train, y_train, X_test, y_test = load_dataset(train_path, test_path)
```

η οποία δέχεται ως είσοδο δύο συμβολοσειρές με τη διαδρομή προς το σύνολο εκπαίδευσης και δοκιμής και εκτελεί τις ακόλουθες ενέργειες

1. Αφαιρεί τις στήλες `fnlwgt` και `education-num` (αριθμητική κωδικοποίηση της στήλης `education`), μιας και δεν προσφέρουν ουσιαστική πληροφορία για την εκτίμησή μας.

2. Αντικαθιστά την κάθε κατηγορική μεταβλητή με εικονικές μεταβλητές (dummy variables). Οι εικονικές μεταβλητές αντικαθιστούν μία κατηγορική μεταβλητή με τόσες δυαδικές μεταβλητές (δηλ. 0 ή 1) όσες είναι και οι κατηγορίες της μεταβλητής. Έτσι αν μία μεταβλητή x_i είναι κατηγορική μεταβλητή με n_i τιμές, τότε αυτή αντικαθίσταται με n_i μεταβλητές $x_{ij} \in \{0, 1\}$, με $j = 1, \ldots, n_i$. Όλες οι μεταβλητές είναι 0, εκτός από αυτή που αντιστοιχεί στην τιμή της x_i, που έχει τιμή 1.

3. Επιστρέφει μία μεταβλητή `X_train` που είναι ένας numpy array που περιέχει τα δεδομένα εκπαίδευσης που προκύπτουν, χωρίς τη μεταβλητή στόχο (δηλ. χωρίς την τελευταία στήλη).

4. Επιστρέφει μία μεταβλητή `y_train` που είναι ένα διάνυσμα (numpy array) που περιέχει τις τιμές της μεταβλητής στόχου.

5. Αντίστοιχα, επιστρέφει δύο μεταβλητές `X_test` και `y_test` με τις τιμές για τις μεταβλητές του συνόλου δοκιμής.

Ερώτημα 1: Εξηγήστε πως προκύπτει η νέα διάσταση (αριθμός χαρακτηριστικών) των δεδομένων σας.

1.2 Κανονικοποίηση δεδομένων

Μετατρέψτε το εύρος τιμών του κάθε χαρακτηριστικού στο $[0, 1]$, τόσο στο σύνολο εκπαίδευσης όσο και στο σύνολο δοκιμής, ως εξής

$$x'_i = \frac{x_i - \min(x_i)}{\max(x_i) - \min(x_i)}$$

όπου $\max(x_i)$ και $\min(x_i)$ είναι η μέγιστη και η ελάχιστη τιμή του i-οστού χαρακτηριστικού στο σύνολο εκπαίδευσης. Μπορείτε να υλοποιήσετε αυτό τον μετασχηματισμό εσείς, ή να χρησιμοποιήσετε την κλάση `MinMaxScaler` του περιβάλλοντος `scikit-learn`. Οι παράμετροι (δηλ. η ελάχιστη και μέγιστη τιμή του κάθε χαρακτηριστικού) πρέπει να εκτιμηθούν μόνο από το σύνολο εκπαίδευσης `X_train` και να εφαρμοστούν τόσο στο σύνολο εκπαίδευσης όσο και στο σύνολο δοκιμής. Στα επόμενα ερωτήματα θα χρησιμοποιήσετε αποκλειστικά τα κανονικοποιημένα σύνολα εκπαίδευσης και δοκιμής.

1.3 Εκτύπωση μετρικών

Υλοποιήστε μία συνάρτηση

```
print_metrics(model, X_train, y_train, X_test, y_test)
```

η οποία δέχεται ένα μοντέλο και τυπώνει (α) την ορθότητα (accuracy), (β) την ακρίβεια (precision) και την ανάκληση (recall) για τη θετική κλάση (>= 50K), (γ) την ακρίβεια (precision) και την ανάκληση (recall) για την αρνητική κλάση (< 50K). Οι τιμές πρέπει να υπολογίζονται και να τυπώνονται τόσο για το σύνολο εκπαίδευσης, όσο και για το σύνολο δοκιμής. Τις μετρικές μπορείτε να τις υπολογίσετε με δικό σας κώδικα ή να καλέσετε τις αντίστοιχες συναρτήσεις του περιβάλλοντος `scikit-learn` (π.χ. την `sklearn.metrics.accuracy_score()`) για την ορθότητα).

1.4 Κατηγοριοποίηση με λογιστική παλινδρόμηση

Υλοποιήστε ένα μοντέλο λογιστικής παλινδρόμησης (ισοδύναμα, έναν perceptron με μία απλή είσοδο και έξοδο) στο περιβάλλον `tensorflow.keras`. Εκπαιδεύστε το μοντέλο στα κανονικοποιημένα δεδομένα εκπαίδευσης χρησιμοποιώντας 10 epochs, με αλγόριθμο βελτιστοποίησης τον Stochastic Gradient Descent με ρυθμό εκμάθησης 0.01. Καλέστε τη συνάρτηση `print_metrics` ώστε να αξιολογήσετε την επίδοσή της.

1.5 Κατηγοριοποίηση με δίκτυο με κρυφά επίπεδα

Όπως και στο προηγούμενο ερώτημα, μόνο που αυτή τη φορά χρησιμοποιήστε ένα μοντέλο με δύο κρυφά επίπεδα με 100 και 200 νευρώνες αντίστοιχα, και 30 epochs για την εκπαίδευση. Στα ενδιάμεσα επίπεδα, χρησιμοποιήστε τη συνάρτηση ενεργοποίησης ReLU.

Ερώτημα 2: Σχολιάστε την επίδοση των δύο μοντέλων. Που πιστεύετε ότι οφείλεται η χειρότερη επίδοση στην θετική κλάση; Ποια θεωρείτε ότι είναι τα πλεονεκτήματα και τα μειονεκτήματα του κάθε μοντέλου;

2 Κατηγοριοποίηση εικόνων με συνελικτικά νευρωνικά δίκτυα

Στο θέμα αυτό θα υλοποιήσετε ένα μοντέλο κατηγοριοποίησης εικόνων στο περιβάλλον `tensorflow.keras`.

2.1 Σύνολο δεδομένων

Το σύνολο δεδομένων θα είναι το CIFAR-10

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

το οποίο σας δίνεται απευθείας μέσα από το περιβάλλον του `tensorflow` καλώντας τη συνάρτηση

```
tf.keras.datasets.cifar10.load_data()
```

Το σύνολο δεδομένων αποτελείται από 50000 εικόνες εκπαίδευσης και 10000 εικόνες δοκιμής, και στόχος είναι η κατηγοριοποίηση των εικόνων σε 10 κατηγορίες αντικειμένων.

2.2 Δημιουργία συνελικτικού δικτύου

Υλοποιήστε μία συνάρτηση

```
cifar_model(lr, alpha)
```

η οποία δέχεται ως ορίσματα τον ρυθμό εκμάθησης `lr` και τον συντελεστή ομαλοποίησης `alpha` και δημιουργεί και επιστρέφει ένα μοντέλο κατηγοριοποίησης εικόνων χρησιμοποιώντας το *Functional API* του keras, σύμφωνα με τις παρακάτω προδιαγραφές:

Αρχιτεκτονική: Το μοντέλο αποτελείται από

1. Ένα συνελικτικό επίπεδο με 16 φίλτρα 3×3 τα οποία επιστρέφουν εικόνες μεγέθους ίσες με την εικόνα εισόδου με συνάρτηση ενεργοποίησης ReLU
2. Ένα επίπεδο συγκέντρωσης 2×2 max pooling
3. Ένα συνελικτικό επίπεδο με 32 φίλτρα 3×3 τα οποία επιστρέφουν εικόνες μεγέθους ίσες με την εικόνα εισόδου με συνάρτηση ενεργοποίησης ReLU
4. Ένα επίπεδο συγκέντρωσης 2×2 max pooling
5. Ένα συνελικτικό επίπεδο με 64 φίλτρα 3×3 τα οποία επιστρέφουν εικόνες μεγέθους ίσες με την εικόνα εισόδου με συνάρτηση ενεργοποίησης ReLU
6. Ένα επίπεδο συγκέντρωσης 2×2 max pooling
7. Ένα επίπεδο εξόδου που εκτιμά την πιθανότητα για καθεμία από τις 10 κατηγορίες

Κάθε επίπεδο εφαρμόζει ομαλοποίηση L_2 στα φίλτρα των συνελικτικών δικτύων, με συντελεστή `alpha`. Το μοντέλο προετοιμάζεται (μέσω της `compile()`) για εκπαίδευση με τον αλγόριθμο βελτιστοποίησης Adam με ρυθμό εκμάθησης `lr` και συνάρτηση απώλειας την κατηγορική διεντροπία (Categorical Cross-entropy).

Ερώτημα 3: Πόσες παραμέτρους εκπαίδευσης έχει το μοντέλο σας; Εξηγήστε.

2.3 Εκπαίδευση μοντέλου

Εκπαιδεύστε το μοντέλο σας χρησιμοποιώντας σύνολο επικύρωσης 20% των δεδομένων εκπαίδευσης, μέγιστο αριθμό 100 εποχών εκπαίδευσης, και πρώιμη διακοπή (early stopping) της εκπαίδευσης αν η απώλεια στο σύνολο επικύρωσης δε μειωθεί για 10 διαδοχικές εποχές. Τα παραπάνω υλοποιούνται εύκολα ως ορίσματα στη συνάρτηση `fit()` του μοντέλου (δείτε τα ορίσματα `epochs`, `callbacks`, `validation_split`), καθώς και την callback κλάση `tf.keras.callbacks.EarlyStopping`.

Ερώτημα 4: Αξιολογήστε το μοντέλο σας για διαφορετικούς συνδυασμούς των παραμέτρων `lr` και `alpha`. Προτείνετε κάποιον συνδυασμό υπερπαραμέτρων. Τι επίδοση έχει το μοντέλο για `lr = 0.001`, `alpha = 0` και πως συγκρίνεται με την επίδοση για `lr = 0.001`, `alpha = 0.01`; Πως εξηγείτε τη διαφορά στην επίδοση των δύο μοντέλων;