Εθνικό Μετσόβιο Πολυτεχνείο

Σχολή Ηλεκτρολόγων Μηχανικών και Μηχανικών Υπολογιστών

Δ.Π.Μ.Σ. "Επιστήμη Δεδομένων και Μηχανική Μάθηση"

ML on FPGAs - Άσκηση 1

Συγγραφείς:
Παναγιώτης Χρονόπουλος 03400240
Ευαγγελία Κοσκινιώτη 03400219
Γιώργος Καλομοίρης 03400215
Ανδρέας Καλογεράς 03400214
Αντώνης Προμπονάς 03400232

Υπεύθυνοι Καθηγητές:
Γκούμας Γεώργιος
Σούντρης Δημήτριος
Κοζύρης Νεκτάριος

Η εργασία κατατέθηκε για το μάθημα:
Παράλληλες Αρχιτεκτονικές Υπολογισμού για Μηχανική Μάθηση

Ιούλιος 2024

1 Εισαγωγή

Στα πλαίσια αυτής της εργασίας κληθήκαμε να εφαρμόσουμε τις δυνατότητες επιτάχυνσης των FPGA σε διάφορες εφαρμογές Μηχανικής Μάθησης, ξεκινώντας από απλούστερες διαδικασίες όπως η πρόσθεση και η αφαίρεση διανυσμάτων και συνεχίζοντας εκπαιδεύοντας μοντέλα Naive Bayes και Logistic Regression. Επιθυμούμε να μελετήσουμε εάν η χρήση των FPGA σημειώνει, σημαντική επιτάχυνση σε όλες τις εφαρμογές σε σχέση με την αντίστοιχη εκτέλεσή τους σε μια απλή CPU και εάν προσφέρει δυνατότητες για περισσότερο υπολογιστικά κοστοβόρες εργασίες. Για αυτόν τον σκοπό θα χρησιμοποιήσουμε την πλατφόρμα **InAccel Coral** μέσω της αντίστοιχης βιβλιοθήκης στη γλώσσα Python.

2 Άθροισμα και Αφαίρεση Διανυσμάτων

Θα ξεκινήσουμε επιταχύνοντας τις εργασίες πρόσθεσης και αφαίρεσης διανυσμάτων. Αρχικά κατανέμουμε 4 τυχαία αρχικοποιημένα διανύσματα (numpy arrays) στη μνήμη του FPGA και στη συνέχεια πραγματοποιούμε αιτήματα για τους υπολογισμούς σε έναν συγκεκριμένο accelerator που έχει σχεδιαστεί για την πρόσθεση και ονομάζεται **vector.addition**. Αφού επιβεβαιώσουμε την ορθότητα της πρόσθεσης την εκτελούμε πρώτα σε CPU και στη συνέχεια σε ένα FPGA. Οι χρόνοι που πήραμε κατά μέση τιμή πάνω σε 7 διαφορετικά runs ήταν:

- **CPU:** $1.21ms \pm 76.4\mu s$
- **FPGA:** $997ns \pm 10.2ns$
- **Addition Speedup = 1210.18**

Επαναλάβαμε την παραπάνω διαδικασία για την αφαίρεση των διανυσμάτων χρησιμοποιώντας τον ίδιο accelerator της πρόσθεσης, αλλά μετατρέποντας το τυχαίο διάνυσμα β της πράξης $\alpha + \beta$ σε αρνητικό. Ξανά οι χρόνοι που πήραμε και το αντίστοιχο speedup ήταν:

- **CPU:** $1.21ms \pm 76.4\mu s$
- **FPGA:** $997ns \pm 10.2ns$
- **Subtraction Speedup = 1141.53**

Βάσει των αποτελεσμάτων των πειραμάτων μας, είναι προφανές ότι η χρήση του FPGA βελτιώνει σημαντικά την υπολογιστική απόδοση σε σχέση με την χρήση μιας απλής CPU, τόσο στην πρόσθεση όσο και στην αφαίρεση διανυσμάτων. Αυτές οι πράξεις αποτελούν βάση για την εκπαίδευση πολλών περίπλοκων δικτύων, επομένως μπορούμε να δούμε εύκολα πως η επιτάχυνσή τους μπορεί να επιφέρει κέρδος πολύτιμου χρόνου και πόρων σε πολλές εφαρμογές μηχανικής μάθησης.

3 Scikit-Learn on FPGAs

3.1 Εφαρμογή σε Naive Bayes

Στη συνέχεια θα εξετάσουμε την εκτέλεση του αλγορίθμου Naive Bayes της βιβλιοθήκης Scikit-Learn όταν αυτή πραγματοποιείται σε CPU και σε FPGA, με διαφορετικό αριθμό δεδομένων προκειμένου να παρατηρήσουμε την επίδρασή του FPGA στο inference ενός πραγματικού αλγορίθμου μηχανικής μάθησης. Σε αυτό το πλαίσιο, αρχικά δημιουργείται ένα σύνολο δεδομένων με 10.000 δείγματα, 500 χαρακτηριστικά και 10 κλάσεις και εκπαιδεύεται ένας ταξινομητής Naive Bayes σε αυτά, με χρήση CPU. Στη συνέχεια εφαρμόζεται το εκπαιδευμένο μοντέλο για να ταξινομήσει ένα ποσοστό των αρχικών δειγμάτων, πρώτα με CPU και στη συνέχεια με FPGA προκειμένου να συγκρίνουμε τους χρόνους του inference ανάμεσα στις δύο συσκευές. Οι χρόνοι που καταγράψαμε ήταν:

- **CPU Classification Time:** $0.39sec$, **FPGA Classification Time:** $0.06sec$
- **Naive Bayes Classification Speedup = 6.5**

Η επιτάχυνση είναι για άλλη μια φορά προφανής και σημαντική. Στη συνέχεια, επαναλαμβάνουμε αυτό το πείραμα πάνω σε ένα σύνολο δεδομένων 100.000 δειγμάτων χρησιμοποιώντας διαφορετικό αριθμό χαρακτηριστικών και κλάσεων, προκειμένου να έχουμε μια καλύτερη εικόνα για την συσχέτιση ανάμεσα σε αυτές τις παραμέτρους και την δυνατότητα της επιτάχυνσης του FPGA. Συγκεκριμένα οι τιμές των παραμέτρων είναι:

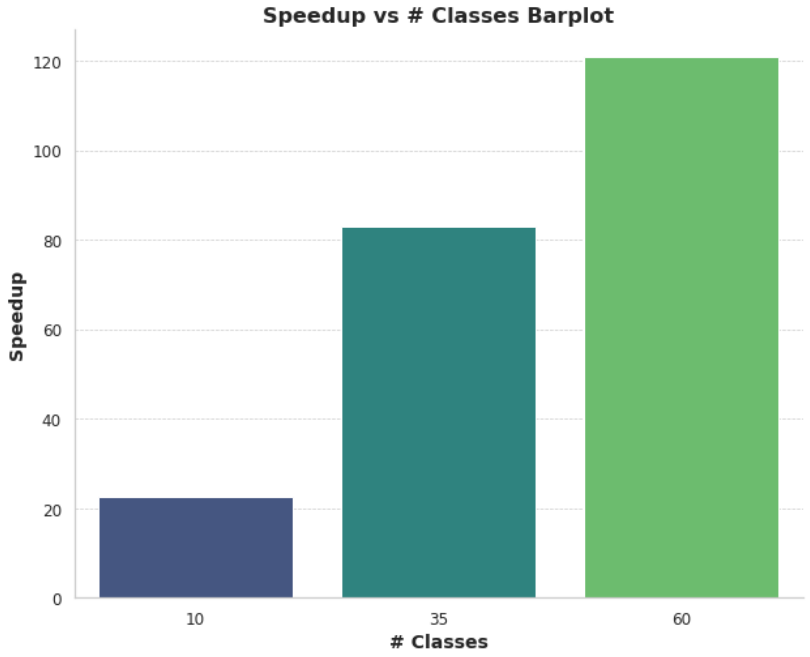
$$samples = [100000]$$
$$features = [500, 1000, 2000]$$
$$classes = [10, 35, 60]$$

Υπολογίζουμε την επιτάχυνση για όλους αυτούς τους συνδυασμούς και παίρνουμε τα παρακάτω αποτελέσματα:

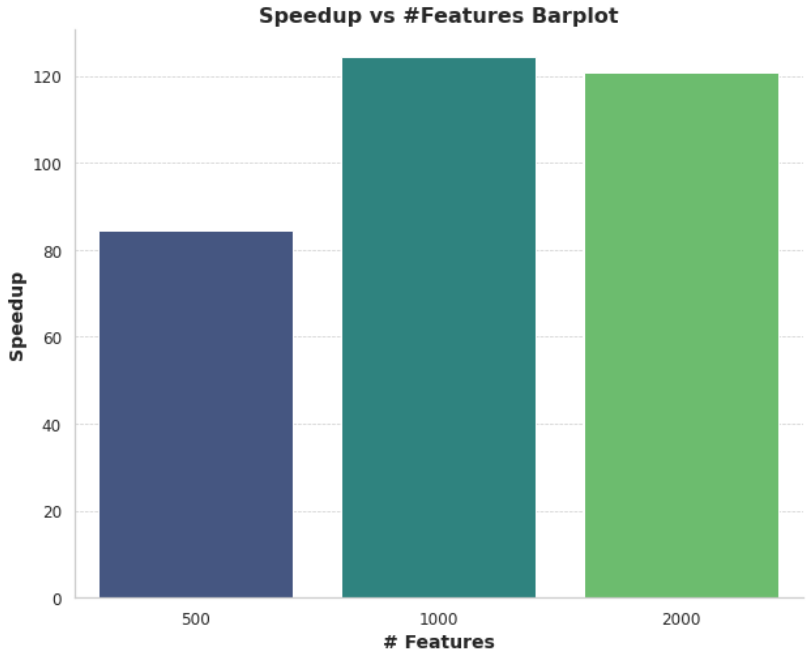
Number of Features	**Number of Classes**	**Speedup**
500	10	20.38
500	35	63.59
500	60	84.42
1000	10	22.31
1000	35	75.32
1000	**60**	**124.34**
2000	10	22.59
2000	35	83.09
2000	60	120.84

Επιτάχυνση Inference Naive Bayes για 100.000 δείγματα

Η μεγαλύτερη επιτάχυνση παρατηρείται στο πείραμα που περιλάμβανε 1000 χαρακτηριστικά και 60 κλάσεις και ήταν ίση με 124.34. Αυτό το αποτέλεσμα είναι λογικό καθώς στο στάδιο του inference, ο Naive Bayes υπολογίζει τις posterior πιθανότητες για κάθε κλάση, κάτι που θα μπορούσε να είναι πολύ κοστοβόρο όσο αυξάνεται ο αριθμός των κλάσεων. Έτσι, αν αυτός ο υπολογισμός γίνεται ανεξάρτητα για κάθε κλάση από το FPGA είναι αναμενόμενο η επιτάχυνση να είναι μέγιστη σε αυτή την περίπτωση, καθώς ο υπολογισμός των posterior είναι πολύ πιο αργός στην CPU, ειδικά όταν ο αριθμός των χαρακτηριστικών είναι μεγάλος. Εξάλλου, η δεύτερη μεγαλύτερη επιτάχυνση προέκυψε για 2000 χαρακτηριστικά και 60 κλάσεις και πολύ παραπλήσια με τη μεγαλύτερη, ίση με 120.84, γεγονός που επιβεβαιώνει το παραπάνω συμπέρασμά μας. Παρατίθενται επίσης δύο bar plots που δείχνουν την επιτάχυνση με βάση τον αριθμό των χαρακτηριστικών για 60 κλάσεις και την επιτάχυνση με βάση τον αριθμό των κλάσεων για 2000 χαρακτηριστικά.



Επιτάχυνση για 2000 χαρακτηριστικά με βάση τον αριθμό των κλάσεων



Επιτάχυνση για 60 κλάσεις με βάση τον αριθμό των χαρακτηριστικών

Εξετάζοντας τα παραπάνω διαγράμματα είναι εμφανής η σχεδόν γραμμική αύξηση της επιτάχυνσης με βάση τον αριθμό των κλάσεων, γεγονός που εξηγήσαμε παραπάνω. Ωστόσο, με την αύξηση του αριθμού των χαρακτηριστικών δεν παρατηρούμε αντίστοιχα δραματική αύξηση της επιτάχυνσης, και μάλιστα μετά τα 1.000 χαρακτηριστικά η επιτάχυνση μειώνεται ελαφρά. Αυτές οι παρατηρήσεις μας οδηγούν στο να πούμε με σιγουριά ότι ο συγκεκριμένος FPGA accelerator είναι καλύτερο να εφαρμοστεί σε ένα σύνολο δεδομένων με πολλές κλάσεις καθώς θα παρέχει γραμμική επιτάχυνση, ενώ σε ένα σύνολο δεδομένων με πολλά χαρακτηριστικά η δυνατότητα

επιτάχυνσης θα συγκλίνει μετά από ένα συγκεκριμένο σημείο και πολύ νωρίτερα από ότι στην περίπτωση των κλάσεων.

4 Logistic Regression

Στην τελευταία ενότητα αυτής της εργασίας θα εξετάσουμε την επίδραση ενός FPGA accelerator στην εκπαίδευση αλλά και τη ρύθμιση των υπερπαραμέτρων ενός μοντέλου μηχανικής μάθησης, και συγκεκριμένα του αλγορίθμου Logistic Regression της Scikit-Learn. Πραγματοποιούμε Grid Search, δηλαδή μια εξαντλητική αναζήτηση πάνω στο σύνολο των παραμέτρων:

$$max\_iters = [50, 100]$$
$$l1\_ratio = [0.3, 0.9]$$

χρησιμοποιώντας πρώτα CPU και στη συνέχεια FPGA, εφαρμόζωντας ταυτόχρονα 3-fold cross validation και παίρνουμε τους εξής χρόνους και βέλτιστες παραμέτρους:

- **CPU Grid Search:** 195 sec training time, 91.64% accuracy, l1_ratio = 0.3, max_iter = 50
- **FPGA Grid Search:** 31 sec training time, 91.66% accuracy, l1_ratio = 0.3, max_iter = 50
- **Speedup = 6.2**

Παρατηρούμε ότι και στις δύο περιπτώσεις καταλήγουμε στο ίδιο σύνολο βέλτιστων υπερπαραμέτρων, **l1_ratio = 0.3, max_iter = 50**, όπως ήταν αναμενόμενο, και πετυχαίνουμε την ίδια ακρίβεια. Ωστόσο, ο χρόνος εκπαίδευσης του grid search στην CPU είναι πολύ μεγαλύτερος από ότι στο FPGA, με αποτέλεσμα να παίρνουμε επιτάχυνση ίση με 6.2. Ειδικά σε διεργασίες που είναι πολύ κοστοβόρες, όπως το grid search, η επιτάχυνση τέτοιας τάξης είναι πολύ σημαντική καθώς μας δίνει τη δυνατότητα να δοκιμάσουμε πολύ περισσότερους συνδυασμούς υπερπαραμέτρων στο ίδιο χρονικό πλαίσιο και έτσι να εκπαιδεύσουμε την καλύτερη δυνατή εκδοχή του μοντέλου σε λογικά χρονικά πλαίσια.