Εθνικο Μετσοβιο Πολυτεχνειο

Σχολη Ηλεκτρολογων Μηχανικων και Μηχανικων Υπολογιστων

Τομεας Τεχνολογιας Πληροφορικης και Υπολογιστων

Βαθιά ενισχυτική μάθηση με χρήση μεθόδων εξερεύνησης βασισμένων σε σήμα ανταμοιβής περιέργειας

Διπλωματικη Εργασια

του

ΜΑΡΙΟΥ Π. ΒΛΑΧΟΓΙΑΝΝΟΠΟΥΛΟΥ

Επιβλέπων: Ανδρέας Γεώργιος Σταφυλοπάτης
Καθηγητής Ε.Μ.Π.

Αθήνα, Οκτώβριος 2021

Εθνικό Μετσόβιο Πολυτεχνείο
Σχολή Ηλεκτρολόγων Μηχανικών και Μηχανικών Υπολογιστών
Τομέας Τεχνολογίας Πληροφορικής και Υπολογιστών

Βαθιά ενισχυτική μάθηση με χρήση μεθόδων εξερεύνησης βασισμένων σε σήμα ανταμοιβής περιέργειας

Διπλωματικη Εργασια

του

ΜΑΡΙΟΥ Π. ΒΛΑΧΟΓΙΑΝΝΟΠΟΥΛΟΥ

Επιβλέπων: Ανδρέας Γεώργιος Σταφυλοπάτης
Καθηγητής Ε.Μ.Π.

Εγκρίθηκε από την τριμελή εξεταστική επιτροπή την 26η Οκτωβρίου 2021.

(Υπογραφή)	*(Υπογραφή)*	*(Υπογραφή)*
.....	.....	.....
Ανδρέας Γεώργιος Σταφυλοπάτης	Στέφανος Κόλλιας	Γιώργος Στάμου
Καθηγητής Ε.Μ.Π.	Καθηγητής Ε.Μ.Π.	Καθηγητής Ε.Μ.Π.

Αθήνα, Οκτώβριος 2021

(Υπογραφή)

.....

Μαριος Βλαχογιαννοπουλος
Διπλωματούχος Ηλεκτρολόγος Μηχανικός και Μηχανικός Υπολογιστών Ε.Μ.Π.

Ευχαριστίες

Θα ήθελα να ευχαριστήσω τον επιβλέποντα καθηγητή κ. Σταφυλοπάτη για την εμπιστοσύνη που μου έδειξε σχετικά με την εκπόνηση της παρούσας εργασίας. Επίσης ευχαριστώ ιδιαίτερα τον κ. Θ. Τασάκο για την καθοδήγηση του και τον κ. Γ. Σιόλα, ο οποίος υπήρξε καταλύτης για την ολοκλήρωση της εργασίας. Τέλος θα ήθελα να ευχαριστήσω την οικογένειά και τους φίλους μου, για την υποστήριξη τους.

Περίληψη

Αντικείμενο της διπλωματικής εργασίας είναι η εξέταση μεθόδων εξερεύνησης μέσω μπόνους περιέργειας στο πλαίσιο αλγορίθμων βαθιάς ενισχυτικής μάθησης σε περιβάλλοντα βιντεοπαιχνιδιών Atari. Το πρόβλημα της εκμετάλλευσης-εξερεύνησης είναι κεντρικό στο πεδίο της ενισχυτικής μάθησης και αφορά την εξισορρόπηση της εκμετάλλευσης των καλύτερων πολιτικών που έχουν βρεθεί να αποδίδουν υψηλές επιστροφές και της εξερεύνησης του χώρου καταστάσεων-δράσεων για ανακάλυψη πιθανώς καλύτερων αποδοχών.

Χρησιμοποιούμε τεχνικές που βασίζονται στη παραγωγή ενός σήματος ανταμοιβής περιέργειας και την υπέρθεση αυτού στο σήμα ανταμοιβής που δίνει το περιβάλλον, με σκοπό να επιτύχουμε εξερεύνηση καταστάσεων που παρουσιάζουν ενδιαφέρον. Το ενδιαφέρον των καταστάσεων μπορεί να μοντελοποιηθεί είτε μέσω της δυναμικής είτε μέσω της καινοτομίας της κατάστασης. Οι μέθοδοι δυναμικής ορίζουν το ενδιαφέρον μιας κατάστασης ως την αδυναμία πρόβλεψης της κατάστασης δεδομένης της προηγούμενης κατάστασης και της δράσης που επιλέχθηκε. Οι μέθοδοι καινοτομίας αφορούν την ανίχνευση καταστάσεων που ο πράκτορας δεν έχει επισκεφθεί στο παρελθόν. Τέλος, μελετάμε την ικανότητα πρακτόρων να επιτύχουν υψηλές επιστροφές χρησιμοποιώντας μόνο το σήμα εξερεύνησης, μηδενίζοντας το σήμα ανταμοιβής που προέρχεται από το περιβάλλον.

Λέξεις Κλειδιά

Δυναμικός Προγραμματισμός, Ενισχυτική Μάθηση, Νευρωνικά Δίκτυα, Πράκτορες Κριτή-Δράστη, Εξερεύνηση, Περιέργεια, Καινοτόμες καταστάσεις, Πρόβλεψη Δυναμικής Περιβάλλοντος

Abstract

The purpose of this diploma thesis is to study exploration methods based on curiosity in the context of deep reinforcement learning with application in Atari video games. The problem of exploration-exploitation is central in the field of reinforcement learning and refers to the balancing between exploiting the already learned best policies to achieve high returns and exploring the state-action space to discover better policies.

We use methods based on the production of a curiosity reward signal that is superimposed on the external reward signal, given by the environment, that attempts to lead the agent to explore interesting states. Curiosity can be modeled either through the dynamics of the environment or through the novelty of a state. For methods using the dynamics of the environment, we consider interesting states as those that the agent is unable to predict given the previous state and the action taken. Novelty based methods attempt to discover states that have not been previously seen by the agent. Finally, we study the ability of agents to achieve high returns, solely using the exploration bonus, without any extrinsic reward signal from the environment.

Keywords

Dynamic Programming, Reinforcement Learning, Neural Networks, Actor-Critic Agents, Exploration, Curiosity, Novel States, Environment Dynamics Prediction

Περιεχόμενα

Κατάλογος Σχημάτων

Κατάλογος Πινάκων

Κεφάλαιο 1

Εισαγωγή

Το πεδίο της ενισχυτικής μάθησης αφορά την εκπαίδευση ενός πράκτορα ώστε να μπορεί να παίρνει αποφάσεις με σκοπό την επίτευξη ενός στόχου. Ο στόχος αυτός αφορά την μεγιστοποίηση ενός σήματος ανταμοιβής που ο πράκτορας δέχεται από το περιβάλλον. Η μοντελοποίηση αυτή αποτελεί μια απλούστευση της διαδικασίας με την οποία βιολογικοί οργανισμοί μαθαίνουν να συμπεριφέρονται στο περιβάλλον τους, σύμφωνα με τη θεωρία της ανταμοιβής και τιμωρίας.

Στο πλαίσιο της ενισχυτικής μάθησης, οι δράσεις του πράκτορα δεν καθορίζονται από κάποιο εκπαιδευτή, αλλά πρέπει μόνος του να μπορεί να πάρει αποφάσεις, δοκιμάζοντας δράσεις και παρατηρώντας τα αποτελέσματα με απώτερο στόχο τη μεγιστοποίηση των συνολικών απολαβών του. Αυτή η διαδικασία απέχει πολύ από το πλαίσιο της επιβλεπόμενης μάθησης, όπου δίνονται οι σωστές απαντήσεις-στόχοι και τα αντίστοιχα συστήματα καλούνται να προσαρμόσουν τις παραμέτρους τους με σκοπό την επίτευξη αυτών των στόχων. Όπως θα δούμε, οι πράκτορες ενισχυτικής μάθησης παράγουν μόνοι τους τους στόχους, κάνοντας εκτιμήσεις σχετικά με το πως λειτουργεί το περιβάλλον τους οι οποίες βασίζονται στα δεδομένα που συλλέγουν, αλληλεπιδρώντας με αυτό.

Η ενισχυτική μάθηση, αντλεί πολλές από τις ιδέες της από το πεδίο του δυναμικού προγραμματισμού. Ο δυναμικός προγραμματισμός παρέχει λύσεις σε προβλήματα βελτιστοποίησης που χαρακτηρίζονται από βέλτιστη υποδομή. Το πρόβλημα διασπάται σε εμφωλευμένα υποπροβλήματα με τη λύση του να παράγεται αναδρομικά από την λύση των υποπροβλημάτων. Βασική διαφορά της ενισχυτικής μάθησης είναι ότι δεν απαιτεί τη γνώση της δυναμικής του περιβάλλοντος καθώς αυτή εκτιμάται από εμπειρίες του πράκτορα. Η σύγχρονη εκδοχή της ενισχυτικής μάθησης χρησιμοποιεί τις εξελίξεις στην εκπαίδευση τεχνητών νευρωνικών δικτύων για τη προσέγγιση βασικών συναρτήσεων που χρησιμοποιούνται για την βελτίωση της διαδικασίας λήψης αποφάσεων.

Μια από τις προκλήσεις στην ενισχυτική μάθηση είναι το trade-off μεταξύ εκμετάλλευσης και εξερεύνησης. Ένας πράκτορας με στόχο την επίτευξη υψηλών αποδοχών πρέπει να επιλέγει δράσεις τις οποίες έχει δοκιμάσει στο παρελθόν και τις οποίες θεωρεί αποδοτικές. Παράλληλα όμως, πρέπει να δοκιμάζει και αποφάσεις που δεν έχει πάρει στο παρελθόν, προς ανακάλυψη πιθανώς καλύτερων εναλλακτικών αποφάσεων. Ο πράκτορας δηλαδή καλείται να ισορροπεί μεταξύ της εκμετάλλευσης της γνώσης που έχει αναπτύξει για το περιβάλλον του σχετικά

με την επίτευξη πιθανών αποδοχών και την δοκιμή εναλλακτικών δράσεων που μπορεί να αποδειχθούν πιο προσοδοφόρες.

Στην παρούσα εργασία, εξετάζουμε βασικούς αλγορίθμους δυναμικού προγραμματισμού, κλασικής ενισχυτικής μάθησης και βαθιάς ενισχυτικής μάθησης. Τέλος, μελετάμε υπό ένα ενιαίο σύστημα αξιολόγησης, την ικανότητα των διαφόρων τεχνικών να ισορροπούν μεταξύ εκμετάλλευσης και εξερεύνησης στο πλαίσιο περιβαλλόντων με πυκνά σήματα ανταμοιβών.

Κεφάλαιο 2

Νευρωνικά Δίκτυα

Τα νευρωνικά δίκτυα [7], [13] είναι ένα υπολογιστικό μοντέλο, εμπνευσμένο από το νευρολογικό σύστημα βιολογικών οργανισμών, που αποτελεί μια απλουστευμένη προσομοίωση της δραστηριότητας του εγκεφάλου. Βασικό δομικό στοιχείο στις περισσότερες αρχιτεκτονικές αποτελεί ο νευρώνας, μια μονάδα υπολογισμού που δέχεται εισόδους τις οποίες επεξεργάζεται για να παράγει μια έξοδο.

Ομάδες τέτοιων νευρώνων μπορούν να χρησιμοποιηθούν για την κατασκευή ενός επιπέδου, στο οποίο όλοι οι νευρώνες δέχονται τις ίδιες εισόδους και παράγουν διαφορετικές εξόδους. Θεωρώντας πολλά τέτοια επίπεδα στη σειρά, όπου οι έξοδοι ενός επιπέδου τίθενται ως είσοδοι στο επόμενο, μπορούμε να δημιουργήσουμε πολύπλοκες αρχιτεκτονικές που ονομάζονται πολυεπίπεδα νευρωνικά δίκτυα. Αν ο αριθμός των επιπέδων είναι μεγάλος τότε μιλάμε για βαθιά νευρωνικά δίκτυα

Η έξοδος ενός νευρωνικού δικτύου αποτελεί μια εκτίμηση μιας συνάρτησης την οποία καλούμαστε να προσεγγίσουμε. Η προσέγγιση γίνεται με παραδείγματα σημείων της συνάρτησης, τα οποία έχουμε στην διάθεση μας και αποτελούν τα δεδομένα εκπαίδευσης. Στις περιπτώσεις μάθησης με εκπαιδευτή(supervised learning), κάθε παράδειγμα i είναι μια δυάδα $(\mathbf{x_i}, \mathbf{y_i})$ της εισόδου και της επιθυμητής εξόδου της συνάρτησης. Οι διαστάσεις του διανύσματος εισόδου ονομάζονται χαρακτηριστικά.

Θέτοντας τα παραδείγματα ως εισόδους σε ένα νευρωνικό δίκτυο και παρατηρώντας την έξοδό του, μπορούμε να την συγκρίνουμε με την επιθυμητή έξοδο από τα δεδομένα εκπαίδευσης και με βάση το λάθος που προκύπτει, να προσαρμόσουμε τον τρόπο που το νευρωνικό δίκτυο επεξεργάζεται τις εισόδους. Η διαδικασία αυτή αποτελεί μια αδρή περιγραφή της διαδικασίας εκπαίδευση ενός νευρωνικού δικτύου από παραδείγματα.

2.1 Ο Νευρώνας

Ο τρόπος με τον οποίο ο νευρώνας επεξεργάζεται τις εισόδους του x_j δίνεται από την εξίσωση:

$$y = \varphi(\sum_j w_j x_j + b) \tag{2.1}$$

Οι παράμετροι w_j ονομάζονται συναπτικά βάρη και η παράμετρος b ονομάζεται πόλωση. Αυτές αποτελούν τις παραμέτρους του νευρώνα και καλούμαστε να τις υπολογίσουμε μέσω της διαδικασίας εκπαίδευσης. Παρατηρούμε ότι η επεξεργασία που εκτελείται από τον νευρώνα αποτελείται από ένα σταθμισμένο άθροισμα των εισόδων του, ακολουθούμενο από μια συνάρτηση φ, η οποία ονομάζεται συνάρτηση ενεργοποίησης.

2.2 Πολυεπίπεδο Νευρωνικό δίκτυο

Όπως είδαμε μπορούμε να οργανώσουμε πολλούς νευρώνες σε επίπεδα, με τις εξόδους ενός επιπέδου να τίθενται ως είσοδοι στο επόμενο επίπεδο. Δεδομένου διανύσματος εισόδου $a^{[l-1]}$, η έξοδος ενός νευρώνα i στο επίπεδο l δίνεται από

$$a_i^{[l]} = \varphi^{[l]}(\sum_j w_{ij}^{[l]} a_j^{[l-1]} + b_i^{[l]}) \tag{2.2}$$

Ή σε διανυσματική μορφή:

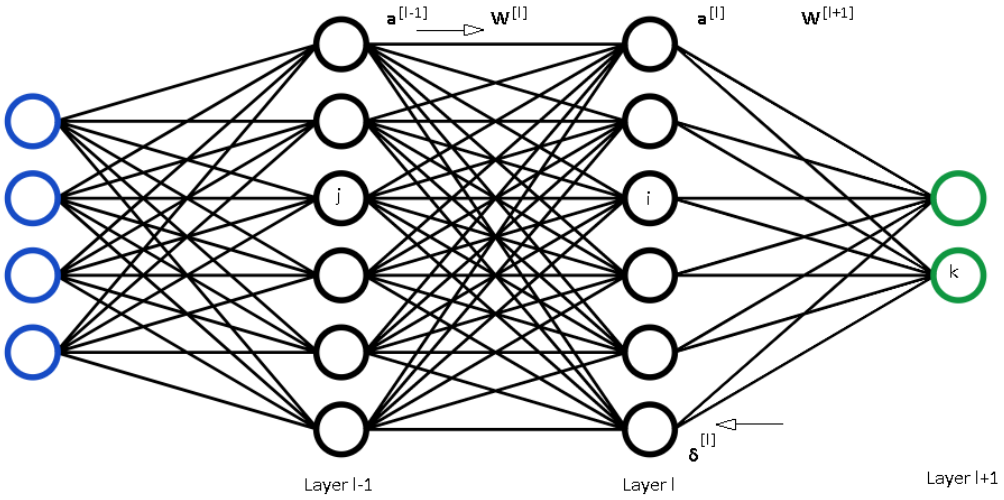
$$\mathbf{a}^{[l]} = \varphi^{[l]}(\mathbf{W}^{[l]}\mathbf{a}^{[l-1]} + \mathbf{b}^{[l]}) \tag{2.3}$$

Όπου $\mathbf{a}^{[l]} \in \mathbb{R}^{n^{[l]}}$ το διάνυσμα εξόδου(ενεργοποίηση) του επιπέδου l, $\mathbf{W}^{[l]} \in \mathbb{R}^{n^{[l]} \times n^{[l-1]}}$ ο πίνακας συναπτικών βαρών w_{ij} μεταξύ του νευρώνα j του επιπέδου $l-1$ και i του επιπέδου l και $n^{[l]}$ το πλήθος των νευρώνων στο επίπεδο l.

Τα ενδιάμεσα επίπεδα ενός δικτύου, δηλαδή τα επίπεδα εξαιρουμένων του πρώτου και τελευταίου, αποτελούν τα κρυφά επίπεδα του δικτύου. Η χρήση τους είναι απαραίτητη για την διαφοροποίηση μεταξύ μη-γραμμικά διαχωρίσιμων δεδομένων[7]. Τα κρυφά επίπεδα δρουν ως ανιχνευτές χαρακτηριστικών(feature detectors). Αυτό σημαίνει ότι κατά την διάρκεια της εκπαίδευσης μαθαίνουν να ανακαλύπτουν τα εξέχοντα χαρακτηριστικά που διαφοροποιούν τα δεδομένα εκπαίδευσης. Αυτό επιτυγχάνεται μέσω μη γραμμικού μετασχηματισμού των δεδομένων εισόδου σε ένα χώρο χαρακτηριστικών, όπου τα προκύπτοντα παραδείγματα είναι γραμμικά διαχωρίσιμα.

Συνάρτηση Λάθους

Η εκπαίδευση ενός νευρωνικού δικτύου συνίσταται στην εύρεση των συναπτικών βαρών και πολώσεων των νευρώνων προς την επίτευξη της επιθυμητής εξόδου για κάθε παράδειγμα του συνόλου δεδομένων. Προς αυτό το σκοπό, πρέπει να οριστεί μια μετρική της απόστασης μεταξύ της επιθυμητής εξόδου και της πραγματικής εξόδου του μοντέλου. Δεδομένης της απόστασης αυτής, η διαδικασία εκπαίδευσης συνίσταται στην ελαχιστοποίηση της απόστασης αυτής για το σύνολο δεδομένων. Η μετρική αυτή ονομάζεται συνάρτηση λάθους και ο ορισμός της εξαρτάται από τα δεδομένα του προβλήματος.



Σχήμα 2.1: Αρχιτεκτονική Δικτύου πολλών επιπέδων

Για παράδειγμα στα πλαίσια ενός προβλήματος παλινδρόμησης, καλούμαστε να εκπαιδεύσουμε ένα μοντέλο να προβλέπει N πραγματικές τιμές εξόδου $\mathbf{y}_j \in \mathbb{R}^N$ για κάθε παράδειγμα j ενός συνόλου δεδομένων με M παραδείγματα. Η συνάρτηση λάθους για ένα παράδειγμα j μπορεί να οριστεί ως η L_2 νόρμα του λάθους μεταξύ της εξόδου $\hat{\mathbf{y}}_j$ και της επιθυμητής εξόδου $\mathbf{y}_j$:

$$\mathcal{L}_j = \|\hat{\mathbf{y}}_j - \mathbf{y}_j\|_2^2 \tag{2.4}$$

Η συνάρτηση κόστους για το σύνολο δεδομένων είναι:

$$\mathcal{J} = \frac{1}{M} \sum_{j=1}^{M} \mathcal{L}_j \tag{2.5}$$

Ένα ακόμη διαδεδομένο πρόβλημα μηχανικής μάθησης είναι το πρόβλημα της ταξινόμησης. Σε αυτό καλούμαστε να μοντελοποιήσουμε μια συνάρτηση που ταξινομεί τα δεδομένα σε επιθυμητές κλάσεις. Η πιο διαδεδομένη προσέγγιση επίλυσης τέτοιων προβλημάτων περιλαμβάνει την μοντελοποίηση, μέσω του νευρωνικού δικτύου, μιας κατανομής πιθανότητας $\hat{\mathbf{y}}$ που εκφράζει την πιθανότητα ένα παράδειγμα να ανήκει σε μια συγκεκριμένη κλάση. Η συνάρτηση λάθους για το παράδειγμα j μπορεί να εκφραστεί με το cross-entropy για c κλάσεις:

$$\mathcal{L}_j = -\sum_{i=1}^{c} y_i^j \log \hat{y}_i^j \tag{2.6}$$

με τη συνάρτηση κόστους να δίνεται από την (2.5)

2.3 Εκπαίδευση πολυεπίπεδων νευρωνικών δικτύων

Για την εύρεση των βαρών και των πολώσεων ενός νευρωνικού δικτύου με χρήση του συνόλου εκπαίδευσης $D = (\mathbf{x}_m, \mathbf{y}_m)$, $m = 1, \ldots, M$, θεωρούμε νευρωνικό δίκτυο L επιπέδων με $n^{[l]}$ νευρώνες ανά επίπεδο l και βάρη $w_{ij}^{[l]}$ και πολώσεις $b_i^{[l]}$. Το βάρος $w_{ij}^{[l]}$ πολλαπλασιάζει την έξοδο του $j-$οστού νευρώνα του επιπέδου $l-1$ και τίθεται ως μια εκ των εισόδων του $i-$οστού νευρώνα του επιπέδου l. Η πόλωση $b_i^{[l]}$ προστίθεται στον γραμμικό συνδυασμό των βαρών $w_{ij}^{[l]}$, $j = 1, \ldots, n^{L-1}$ και των εισόδων στο νευρώνα i και το άθροισμα περνάει από συνάρτηση ενεργοποίησης $\varphi^{[l]}$.

$$\begin{aligned} z_i^{[l]} &= \sum_{j=1}^{n^{[l-1]}} w_{ij}^{[l]} a_j^{[l-1]} + b_i^{[l]} \\ a_i^{[l]} &= \varphi^{[l]}(z_i^{[l]}) \end{aligned} \tag{2.7}$$

Ορίζουμε την συνάρτηση κόστους για το σύνολο εκπαίδευσης

$$\mathcal{J}(\theta) = \frac{1}{M} \sum_{m=1}^{M} \mathcal{L}_m(\hat{y}_m, y_m) \tag{2.8}$$

όπου $\mathcal{L}_m(\hat{y}_m, y_m)$ η συνάρτηση λάθους για το παράδειγμα m και θ το σύνολο παραμέτρων (συναπτικών βαρών και πολώσεων) του δικτύου. Σκοπός της εκπαίδευσης είναι η εύρεση των παραμέτρων θ^* του δικτύου που ελαχιστοποιούν την συνάρτηση κόστους:

$$\theta^* = \underset{\theta}{\operatorname{argmin}} \mathcal{J}(\theta) \tag{2.9}$$

Προς αυτό το σκοπό, αρχικοποιούμε τυχαία τα βάρη και τις πολώσεις και χρησιμοποιούμε τον αλγόριθμο της κατάβασης μέγιστης κλίσης([7]), ανανεώνοντας την τιμές προς την κατεύθυνση της μέγιστης κλίσης.

$$\Delta\theta = -\alpha \frac{\partial \mathcal{J}}{\partial \theta} \tag{2.10}$$

Η υπερπαράμετρος α ονομάζεται ρυθμός μάθησης(learning rate) και καθορίζει το μέγεθος της μεταβολής που επιφέρουμε στις παραμέτρους του δικτύου. Για πολύ μικρές τιμές, το σημείο θ προσεγγίζει αργά το τοπικό ελάχιστο της συνάρτησης J και η εκπαίδευση του δικτύου είναι πολύ αργή. Αντίθετα, πολύ μεγάλες τιμές του ρυθμού εκμάθησης, ενδέχεται να επιφέρουν overshoot του τοπικού ελαχίστου και ταλάντωση γύρω από αυτό ή ακόμα και απόκλιση από αυτό. Ο ρυθμός εκμάθησης είναι γενικά η πιο σημαντική παράμετρος για την επιτυχία της εκπαίδευσης και χρήζει προσεκτικής ρύθμισης.

Για τον υπολογισμό των μερικών παραγώγων $\frac{\partial \mathcal{J}}{\partial \theta}$ ως προς όλες τις παραμέτρους θ του δικτύου, χρησιμοποιούμε τον αλγόριθμο της οπισθοδιάδοσης που βασίζεται στον κανόνα της αλυσίδας παραγώγισης: για διανύσματα $\mathbf{x} \in \mathbb{R}^m$, $\mathbf{y} \in \mathbb{R}^n$ και συναρτήσεις $g : \mathbb{R}^m \to \mathbb{R}^n$, $f : \mathbb{R}^n \to \mathbb{R}$ με $\mathbf{y} = g(\mathbf{x})$ και $z = f(\mathbf{y})$, έχουμε:

$$\frac{\partial z}{\partial x_i} = \sum_j \frac{\partial z}{\partial y_j}\frac{\partial y_j}{\partial x_i} \tag{2.11}$$

Θεωρούμε την (2.10) για τα συναπτικά βάρη και πολώσεις:

$$\begin{aligned} \Delta w_{ij}^{[l]} &= -\alpha \frac{\partial \mathcal{J}}{\partial w_{ij}^{[l]}} \\ \Delta b_i^{[l]} &= -\alpha \frac{\partial \mathcal{J}}{\partial b_i^{[l]}} \end{aligned} \tag{2.12}$$

Από τον κανόνα της αλυσίδας και την (2.7) έχουμε:

$$\begin{aligned} \Delta w_{ij}^{[l]} &= -\alpha \frac{\partial \mathcal{J}}{\partial z_i^{[l]}}\frac{\partial z_i^{[l]}}{\partial w_{ij}^{[l]}} = -\alpha \delta_i^{[l]} a_j^{[l-1]} \\ \Delta b_i^{[l]} &= -\alpha \frac{\partial \mathcal{J}}{\partial z_i^{[l]}}\frac{\partial z_i^{[l]}}{\partial b_i^{[l]}} = -\alpha \delta_i^{[l]} \end{aligned} \tag{2.13}$$

όπου ορίσαμε $\delta_i^{[l]} = \frac{\partial \mathcal{J}}{\partial z_i^{[l]}}$, που ονομάζεται σήμα διόρθωσης. Ο υπολογισμός της $\delta_i^{[l]}$ γίνεται εύκολα για το επίπεδο εξόδου L:

$$\delta_i^{[L]} = \frac{\partial \mathcal{J}}{\partial z_i^{[L]}} = \frac{\partial \mathcal{J}}{\partial a_i^{[L]}}\frac{\partial a_i^{[L]}}{\partial z_i^{[L]}} = \frac{\partial \mathcal{J}}{\partial a_i^{[L]}}\varphi'^{[L]}(z_i^{[L]}) \tag{2.14}$$

όπου φ' η παράγωγος της συνάρτησης φ. Ο υπολογισμός της $\frac{\partial \mathcal{J}}{\partial a_i^{[L]}}$ προκύπτει εύκολα από τον ορισμό της συνάρτησης J. Για τα υπόλοιπα επίπεδα μπορούμε να υπολογίσουμε τα $\delta_i^{[l]}$ συναρτήσει των $\delta_k^{[l+1]}$, $k = 1, ..., n^{[l+1]}$ χρησιμοποιώντας τον κανόνα της αλυσίδας (2.11) και την (2.7):

$$\delta_i^{[l]} = \frac{\partial \mathcal{J}}{\partial z_i^{[L]}} = \sum_{k=1}^{n^{[l+1]}} \frac{\partial \mathcal{J}}{\partial z_k^{[l+1]}}\frac{\partial z_k^{[l+1]}}{\partial z_i^{[l]}} = \sum_{k=1}^{n^{[l+1]}} \delta_k^{[l+1]} w_{ki}^{[l+1]} \varphi'^{[l]}(z_i^{[l]}) \tag{2.15}$$

Ο αλγόριθμος της οπισθοδιάδοσης εκφράζεται σε μητρική μορφή των (2.7), (2.13), (2.14), (2.15):

- Εμπρόσθιο πέρασμα:

$$\begin{aligned} \mathbf{z}^{[l]} &= \mathbf{W}^{[l]}\mathbf{a}^{[l-1]} + \mathbf{b}^{[l]} \\ \mathbf{a}^{[l]} &= \varphi^{[l]}(\mathbf{z}^{[l]}) \end{aligned} \tag{2.16}$$

- Για έξοδο:

$$\begin{aligned} \boldsymbol{\delta}^{[L]} &= \boldsymbol{\nabla}_{a^{[L]}}\mathcal{J} \cdot \varphi'^{[l]}(\mathbf{z}^{[L]}) \\ \Delta\mathbf{W}^{[L]} = -\alpha\boldsymbol{\delta}^{[L]} \cdot \mathbf{a}^{[L-1]} &\ , \ \Delta\mathbf{b}^{[L]} = -\alpha\boldsymbol{\delta}^{[L]} \end{aligned} \tag{2.17}$$

• Οπίσθιο πέρασμα:

$$\begin{aligned} \boldsymbol{\delta}^{[l]} &= [\mathbf{W}^{[l+1]}]^T \boldsymbol{\delta}^{[l+1]} \cdot \varphi'^{[l]}(\mathbf{z}^{[l]}) \\ \Delta\mathbf{W}^{[l]} &= -\alpha\boldsymbol{\delta}^{[l]} \cdot \mathbf{a}^{[l-1]} \ , \ \Delta\mathbf{b}^{[l]} = -\alpha\boldsymbol{\delta}^{[l]} \end{aligned} \tag{2.18}$$

Συνοψίζοντας, ο αλγόριθμος της οπισθοδιάδοσης αποτελείται από την εμπρόσθια τροφοδότηση των παραδειγμάτων μέσω των επιπέδων του δικτύου, με σκοπό τον υπολογισμό της συνάρτησης κόστους για το σύνολο των παραδειγμάτων, ακολουθούμενο από την οπίσθια διάδοση ενός σήματος διόρθωσης των βαρών και των πολώσεων του δικτύου. Η διαδικασία αυτή επαναλαμβάνεται για πλήθος εποχών. Ως εποχή ορίζεται ένας κύκλος της εκπαίδευσης στον οποίο έχουν παρουσιαστεί όλα τα παραδείγματα ακριβώς μια φορά. Συνεπώς, μια επανάληψη του παραπάνω αλγορίθμου συμπίπτει με μια εποχή.

2.4 Λεπτομέρειες Εκπαίδευσης

2.4.1 Γενίκευση

Αποδεικνύεται[7] ότι ένα νευρωνικό δίκτυο με ένα κρυφό επίπεδο και συνεχή συνάρτηση ενεργοποίησης μπορεί να προσεγγίσει οποιαδήποτε συνάρτηση χρησιμοποιώντας ένα σύνολο δεδομένων της με οσοδήποτε ακρίβεια.

Ο κύριος λόγος που στη πράξη χρησιμοποιούνται περισσότερα του ενός κρυφά επίπεδα έχει να κάνει με την ικανότητα γενίκευσης του δικτύου. Θεωρώντας το πρόβλημα της εκπαίδευσης ενός δικτύου ως ένα πρόβλημα προσαρμογής καμπύλης, τότε το δίκτυο μπορεί να ειδωθεί ως μια αντιστοίχιση εισόδων-εξόδων. Ένα δίκτυο λέγεται ότι γενικεύει, όταν το λάθος για ένα παράδειγμα πάνω στο οποίο δεν έχει εκπαιδευτεί είναι μικρό. Ένα δίκτυο που γενικεύει καλά, θα παράγει σωστή αντιστοίχιση μεταξύ εισόδων-εξόδων ακόμα και για εισόδους που δεν έχουν χρησιμοποιηθεί για την εκπαίδευσή του. Εναλλακτικά, η συνάρτηση λάθους θα είναι μικρή τόσο για τα παραδείγματα πάνω στα οποία εκπαιδεύτηκε όσο και σε παραδείγματα που δεν έχει δει.

Αντίθετα, υπάρχει περίπτωση η συνάρτηση λάθους για τα δεδομένα εκπαίδευσης να είναι μικρή, αλλά το λάθος για νέα παραδείγματα να είναι μεγάλο. Ένα τέτοιο δίκτυο δεν είναι ικανό να γενικεύσει και λέμε ότι πάσχει από υπερπροσαρμογή(overfitting) στα δεδομένα εκπαίδευσης. Συνήθως σε αυτές τις περιπτώσεις, το δίκτυο τείνει να μαθαίνει το θόρυβο που υπάρχει στα δεδομένα.

Τέλος, αν μετά την διαδικασία εκπαίδευσης, το λάθος είναι υψηλό τόσο για τα δεδομένα εκπαίδευσης όσο και για δεδομένα που δεν έχουν παρουσιαστεί στο δίκτυο, τότε μιλάμε για υποπροσαρμογή(underfitting). Αυτό το φαινόμενο οφείλεται συνήθως στη χαμηλή πολυπλοκότητα της αρχιτεκτονικής του δικτύου, δηλαδή το μικρό αριθμό επιπέδων ή νευρώνων ανά επίπεδο, καθώς το μοντέλο δεν είναι αρκετό για να διαχωρίσει τα δεδομένα.

2.4.2 Mini-batch, Στοχαστική κατάβαση μέγιστης κλίσης

Συχνά, αντί για την παρουσίαση όλων των παραδειγμάτων του συνόλου εκπαίδευσης για τον υπολογισμό της συνάρτησης κόστους, επιλέγεται η προσέγγιση της συνάρτησης κόστους μέσω του μέσου λάθους για ένα μέρος του συνόλου εκπαίδευσης [13]. Η διαδικασία αυτή ονομάζεται Batch Gradient Descent και επιτυγχάνει ένα trade-off μεταξύ ταχύτητας εκπαίδευσης και ακρίβειας στην προσέγγιση της συνάρτησης κόστους, καθώς κάνουμε updates με μικρότερο υπολογιστικό κόστος αλλά εισάγοντας λάθη στη προσέγγιση της συνάρτησης κόστους. Μια εποχή του Batch Gradient Descent αποτελείται από διαχωρισμό του συνόλου δεδομένων σε ένα πλήθος mini-batches και διαδοχικά updates των παραμέτρων του δικτύου για κάθε batch.

Η περίπτωση όπου στο κάθε mini-batch έχουμε ακριβώς ένα παράδειγμα ονομάζεται στοχαστική κατάβαση μέγιστης κλίσης (SGD) και προσεγγίζουμε την συνάρτηση κόστους με ένα μόνο παράδειγμα.

2.4.3 Συνάρτηση Ενεργοποίησης

Η συνάρτηση ενεργοποίησης ενός νευρώνα εκφράζει αν ο νευρώνας είναι ενεργός για μια συγκεκριμένη είσοδο. Έχουν προταθεί πολλές συναρτήσεις ενεργοποίησης, καθεμιά με τα πλεονεκτήματα και τα μειονεκτήματά της.

Μια ιστορικά δημοφιλής οικογένεια συναρτήσεων ενεργοποίησης είναι οι υπερβολικές και λογιστικές συναρτήσεις με κύρια παραδείγματα την υπερβολική εφαπτομένη και τη σιγμοειδή:

$$\varphi(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}} \tag{2.19}$$

$$\varphi(x) = \frac{1}{1 + e^{-x}} \tag{2.20}$$

Η σιγμοειδής και η υπερβολική εφαπτομένη περιορίζουν την είσοδό τους στα διαστήματα $[0, 1]$, $[-1, 1]$ αντίστοιχα, ενώ οι κλίση τους περιορίζεται στο $[0, 1]$. Αυτό μπορεί να οδηγήσει σε κλίσεις που εξαφανίζονται(vanishing gradients)[13], λόγω της οπισθοδιάδοσης των κλίσεων σε δίκτυα με πολλά επίπεδα, με αποτέλεσμα πολύ μικρά updates και άρα επιβράδυνση της μάθησης.

Για αυτό το λόγο, έχει επικρατήσει να χρησιμοποιείται η συνάρτηση ReLU(Rectified Linear Unit)

$$\varphi(x) = \begin{cases} x, & \text{αν } x > 0 \\ 0, & \text{αλλιώς} \end{cases} \tag{2.21}$$

Το πρόβλημα εξαφάνισης κλίσεων περιορίζεται καθώς οι κλίση της relu είναι είτε 1 είτε 0, και έτσι οι κλίσεις δεν φθίνουν με την οπισθοδιάδοση.

$$\varphi'(x) = \begin{cases} 1, & \text{αν } x > 0 \\ 0, & \text{αλλιώς} \end{cases} \tag{2.22}$$

Παρόλα αυτά η relu αντιμετωπίζει το πρόβλημα ότι ενδέχεται να αχρηστεύσει νευρώνες οι οποίοι δεν θα ενεργοποιηθούν ποτέ. Για αυτό συχνά χρησιμοποιούνται εναλλακτικές όπως η leaky ReLU, με μικρή σταθερά $\alpha > 0$:

$$\varphi(x) = \begin{cases} x, & \text{αν } x > 0 \\ ax, & \text{αλλιώς} \end{cases} \tag{2.23}$$

Σε περιπτώσεις όπου θέλουμε η έξοδος ενός επιπέδου K νευρώνων να είναι μια κατανομή πιθανότητας, όπως τα προβλήματα κατηγοριοποίησης, χρησιμοποιείται η συνάρτηση softmax:

$$\varphi(\mathbf{x})_i = \frac{e^{x_i}}{\sum_{i=1}^{K} e^{x_i}} \tag{2.24}$$

2.4.4 Σύγκλιση και τερματισμός

Η σύγκλιση του αλγορίθμου μέγιστης κλίσης κατάβασης στο ολικό ελάχιστο της συνάρτησης $\mathcal{J}$ εγγυάται σε περιπτώσεις όπου η συνάρτηση $\mathcal{J}$ είναι κυρτή και διαφορίσιμη. Αυτή η υπόθεση όμως δεν ισχύει γενικά στη περίπτωση πολυεπίπεδων νευρωνικών δικτύων. Γενικά, ο αλγόριθμος οπίσθιας διάδοσης δεν μπορεί να αποδειχθεί ότι συγκλίνει και δεν υπάρχουν καλά ορισμένα κριτήρια για τον τερματισμό του[7].

Κάποια κριτήρια τερματισμού που χρησιμοποιούνται συχνά είναι ο τερματισμός όταν η L_2 νόρμα του διανύσματος κλίσης είναι μικρότερη από ένα κατώφλι, ή όταν ο απόλυτος ρυθμός μεταβολής του μέσου κόστους ανά εποχή είναι μικρότερος από κάποιο κατώφλι.

Η πιο διαδεδομένη μέθοδος τερματισμού περιλαμβάνει το διαχωρισμό του συνόλου δεδομένων σε δεδομένα εκπαίδευσης και δεδομένα validation. Το δίκτυο εκπαιδεύεται πάνω στα δεδομένα εκπαίδευσης για ένα πλήθος εποχών και στη συνέχεια ελέγχεται για την απόδοσή στα δεδομένα τεστ, ώστε να διασφαλιστεί η ικανότητα γενίκευσης μέσω της υψηλής απόδοσης στα δύο σύνολα δεδομένων. Η διαδικασία συνεχίζεται, μέχρι το κόστος στο σύνολο validation να γίνει μικρότερο από ένα κατώφλι. Η διαδικασία αυτή μπορεί επίσης να χρησιμοποιηθεί για την ρύθμιση των υπερ-παραμέτρων του δικτύου, όπως την αρχιτεκτονική ή το ρυθμό μάθησης.

2.5 Κατάβασης μέγιστης κλίσης με αδράνεια

Κατά καιρούς, έχουν προταθεί [17] μέθοδοι και βελτιώσεις για τον αλγόριθμο κατάβασης μέγιστης κλίσης.

2.5.1 Αδράνεια

Όπως είδαμε, ο αλγόριθμος μέγιστης κλίσης βρίσκει τις παραμέτρους που επιτυγχάνουν τοπικό ελάχιστο σε κυρτές συναρτήσεις κόστους. Ακόμη [17] μπορούν να βρεθούν σαγματικά σημεία για μη κυρτές συναρτήσεις. Ως σαγματικά σημεία ορίζονται τα σημεία όπου η κλίση της συνάρτησεις μηδενίζεται χωρίς να είναι απαραίτητα τοπικά ελάχιστα ή μέγιστα. Αυτό εμφανίζεται ως ένα πρόβλημα εξαφάνισης κλίσης(vanishing gradient).

Μια λύση που χρησιμοποιείται συχνά είναι η εισαγωγή ενός όρου αδράνειας στις μεταβολές των παραμέτρων του δικτύου. Γενικά, σε περιοχές όπου έχουμε μικρή κλίση, οι παράμετροι ανανεώνονται λίγο ενώ σε απότομες κλίσεις οι μεταβολές είναι μεγάλες. Θέλουμε να αυξήσουμε το ρυθμό μάθησης στις πρώτες περιοχές αλλά όχι τόσο πολύ ώστε να δημιουργείται αστάθεια όταν φτάσουμε στις δεύτερες. Η ιδέα είναι να διατηρούμε ένα εκθετικά σταθμισμένο μέσο όρο των μεταβολών στις παραμέτρους και να ανανεώνουμε τις παραμέτρους προς το μέσο όρο των κλίσεων.

Συγκεκριμένα, δεδομένων $\Delta\theta(t)$ όπως υπολογίστηκαν από το αλγόριθμο οπισθοδιάδοσης (2.16), (2.17), (2.18) τη χρονική στιγμή t για παράμετρο θ, διατηρούμε τη τρέχουσα αδράνεια $m(t)$ και ανανεώνουμε τις παραμέτρους του δικτύου σύμφωνα με την:

$$\begin{aligned} m(t) &= \beta m(t-1) + (1-\beta)\Delta\theta(t) \\ \theta(t+1) &= \theta(t) - \alpha m(t) \end{aligned} \tag{2.25}$$

Αυτή η διαδικασία μπορεί να ειδωθεί ως φιλτράρισμα της μεταβολής $\Delta\theta(t)$ μέσα από βαθυπερατό φίλτρο με συνάρτηση μεταφοράς $H(z) = (1-\beta)/(1-\beta z^{-1})$ με $0 < \beta < 1$ που μειώνει τις ταλαντώσεις που προκύπτουν από ακραίες μεταβολές της κλίσης της συνάρτησης κόστους. Η συνάρτηση $m(t)$ αποτελεί τον όρο αδράνειας και το β ονομάζεται συντελεστής αδράνειας, το οποίο για μεγάλες τιμές επιτρέπει μακροχρόνιους μέσους όρους των κλίσεων.

2.5.2 Ο αλγόριθμος Adam

Ο πιο διαδεδομένος αλγόριθμος είναι ο Adam [17], [9]. Ο αλγόριθμος κρατά τρέχουσες εκθετικά σταθμισμένες εκτιμήσεις της ροπής πρώτης τάξης της μεταβολής του update $m(t)$ με εκθέτη β_m, όπως στη περίπτωση της αδράνειας και της ροπής δεύτερης τάξης $s(t)$ με εκθέτη β_s. Η αρχικοποίηση αυτών των ροπών στο 0 εισάγει μεροληψία προς το 0, η οποία διορθώνεται από τους όρους $\tilde{m}(t)$, $\tilde{s}(t)$.

$$\begin{aligned} &m(t) = \beta_m m(t-1) + (1-\beta_m)\Delta\theta(t) \text{ , ροπή κλίσης 1ης τάξης} \\ &s(t) = \beta_s s(t-1) + (1-\beta_s)(\Delta\theta(t))^2 \text{ , ροπή κλίσης 2ης τάξης} \\ &\tilde{m}(t) = \frac{m(t)}{1-\beta_m^t} \text{ , διόρθωση bias 1ης τάξης} \\ &\tilde{s}(t) = \frac{s(t)}{1-\beta_s^t} \text{ , διόρθωση bias 2ης τάξης} \\ &\theta(t+1) = \theta(t) - \alpha\frac{\tilde{m}(t)}{\epsilon + \sqrt{\tilde{s}}} \text{ , update} \end{aligned} \tag{2.26}$$

2.6 Συνελικτικά Νευρωνικά δίκτυα

Τα συνελικτικά νευρωνικά δίκτυα [10] είναι μια ειδική κατηγορία αρχιτεκτονικών, εμπνευσμένη από το σύστημα όρασης βιολογικών οργανισμών, η οποία χρησιμοποιείται ευρέως για δεδομένα στα οποία παρουσιάζεται χωρική συσχέτιση μεταξύ των χαρακτηριστικών, όπως για παράδειγμα εικόνες. Πρόκειται για πολυεπίπεδα νευρωνικά δίκτυα που είναι σχεδιασμένα να

αναγνωρίζουν δισδιάστατα σχήματα διατηρώντας το αναλλοίωτο αναφορικά με την μετατόπιση, τη κλιμάκωση, τη στρέβλωση, και άλλες μορφές παραμορφώσεων[10].

Η κύρια διαφορά από τα πολυεπίπεδα νευρωνικά δίκτυα που εξετάσαμε προηγούμενα είναι ότι αντικαθίσταται ο γραμμικός συνδυασμός βαρών και εισόδων από το γραμμικό φιλτράρισμα (συνέλιξη) της εισόδου μέσα από παραμετροποιημένα και εκπαιδεύσιμα φίλτρα.

Η αδυναμία των πολυεπίπεδων νευρωνικών δικτύων να διαχειρίζονται δεδομένα, όπως εικόνες, έγκειται εν μέρει στο μεγάλο αριθμό εισόδων(pixel) ο οποίος αυξάνει τον αριθμό των βαρών ακόμα και για μικρό αριθμό νευρώνων ανά επίπεδο. Η βασική δυσκολία όμως, αφορά το γεγονός ότι δεν προσφέρουν κάποιο μηχανισμό που διατηρεί το αναλλοίωτο στη μετατόπιση ή τη κλιμάκωση χαρακτηριστικών, όπως ακμές και γωνίες, στην εικόνα. Αυτό έχει ως αποτέλεσμα, πολλοί νευρώνες να μαθαίνουν παρόμοια χαρακτηριστικά με την διαφορά ότι θα βρίσκονται μετατοπισμένα στην εικόνα. Τέλος, ένα σημαντικό μειονέκτημα της χρήσης νευρωνικών δικτύων για την επεξεργασία εικόνων είναι ότι αγνοούν πλήρως την τοπολογία των εισόδων, η οποία μπορεί να φανεί ιδιαίτερα χρήσιμη στην διαδικασία μάθησης.

Η χρήση της συνέλιξης, μπορεί να εξηγηθεί αν θεωρήσουμε εισόδους 2Δ εικόνες $X[i,j]$ και τα νευρωνικά δίκτυα που έχουμε περιγράψει στις προηγούμενες ενότητες, με την διαφορά ότι οι νευρώνες ενός επιπέδου είναι διατεταγμένοι σε 2Δ δομή. Το βάρος W_{klij} πολλαπλασιάζει τη τιμή του pixel (i,j) και το αποτέλεσμα αποτελεί την έξοδο του pixel (k,l) του επιπέδου αφού προστεθεί στη πόλωση B_{kl}

$$Z[k,l] = B_{kl} + \sum_i \sum_j W_{klij} X[i,j] = B_{kl} + \sum_a \sum_b W_{klab} X[k+a, l+b] \tag{2.27}$$

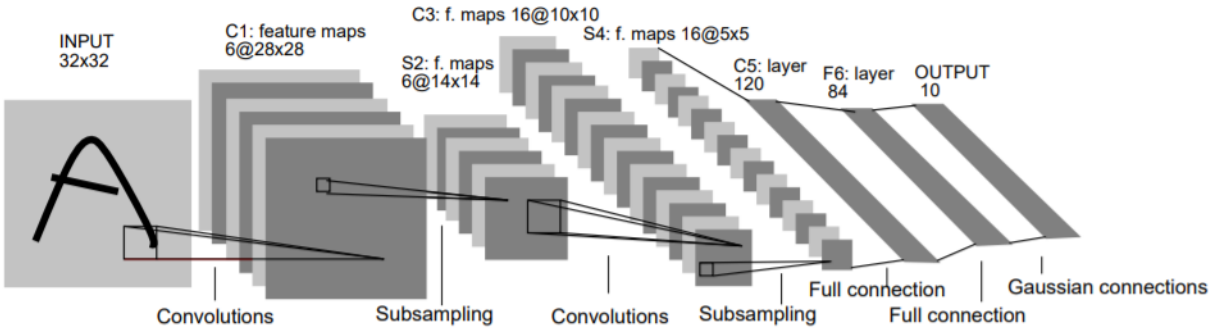
Για να επιτύχουμε το αναλλοίωτο στην μετατόπιση, πρέπει μια μετατόπιση της εικόνας να φέρει την ίδια μετατόπιση στην έξοδο του επιπέδου. Αυτό συμβαίνει μόνο αν τα βάρη και οι πολώσεις W_{klab}, B_{kl} δεν εξαρτώνται από τα k,l.

$$Z[k,l] = B + \sum_a \sum_b W_{ab} X[k+a, l+b] \tag{2.28}$$

Η (2.28) αποτελεί τη σχέση της ετεροσυσχέτισης, η οποία χρησιμοποιείται στην πράξη αντί της συνέλιξης. Επίσης, καταδεικνύει ότι ο αριθμός των παραμέτρων μειώνεται, αφού τα βάρη μοιράζονται από όλα τα pixel εξόδου. Συνεπώς, τα βάρη W_{ab}, B είναι σε θέση να ανιχνεύσουν το ίδιο χαρακτηριστικό οπουδήποτε στην εικόνα εισόδου.

Οι δείκτες (a,b) κινούνται σε μια περιοχή γύρω από το (k,l) εξάγοντας τοπικα χαρακτηριστικά. Η επιλογή αυτή εκφράζει την έννοια του τοπικού δεκτικού πεδίου, η οποία μπορεί να παραλληλιστεί με παρόμοιες διαδικασίες που λαμβάνουν χώρα στο οπτικό φλοιό βιολογικών οργανισμών.

Ένα επίπεδο ενός συνελικτικού δικτύου (Σχήμα 2.2) αποτελείται από ένα πλήθος χαρτών χαρακτηριστικών. Κάθε χάρτης είναι οργανωμένος σε 2 διαστάσεις με τις επιμέρους μονάδες του να μοιράζονται τα ίδια βάρη, και είναι επιφορτισμένος με την εξαγωγή ενός τύπου χαρακτηριστικών. Έτσι, οι μονάδες ενός χάρτη επεξεργάζονται διαφορετικές περιοχές της εικόνας εισόδου με τον ίδιο τρόπο, εξάγοντας το ίδιο χαρακτηριστικό σε όλες τις πιθανές περιοχές



Σχήμα 2.2: Αρχιτεκτονική Συνελικτικού Δικτύου LeNet-5 (Πηγή: [10])

της εικόνας. Οι μονάδες ενός άλλου χάρτη μοιράζονται και αυτές τα δικά τους βάρη και πολώσεις και εξάγουν ένα διαφορετικό τοπικό χαρακτηριστικό. Σημειώνεται επίσης ότι λόγω της συνέλιξης, τα τοπικά δεκτικά πεδία μερικώς επικαλύπτονται.

Όπως είδαμε η χρήση της συνέλιξης έχει το αποτέλεσμα ότι μετατοπίσεις στην εικόνα να επιφέρουν όμοιες μετατοπίσεις στο χάρτη χαρακτηριστικών. Αυτό σημαίνει ότι αν ένα χαρακτηριστικό έχει ανιχνευθεί, τότε η ακριβής θέση του δεν είναι πλέον τόσο σημαντική, όσο η σχετική θέση του με άλλα χαρακτηριστικά. Ένας τρόπος να μειωθεί η ακρίβεια με την οποία κωδικοποιούνται διακριτά χαρακτηριστικά στους χάρτες είναι η μείωση της ανάλυσης της εικόνας μέσω υποδειγματοληψίας.

Τα επίπεδα δειγματοληψίας εκτελούν ένα τοπικό σταθμισμένο μέσο όρο με εκπαιδευσιμες παραμέτρους(ή μέγιστο χωρίς την ανάγκη για επιπλέον παραμέτρους) μειώνοντας την ανάλυση του χάρτη χαρακτηριστικών και την ευαισθησία της εξόδου σε μεταβολές και παραμορφώσεις. Έτσι, προχωρώντας στα επόμενα επίπεδα του δικτύου, το δεκτικό πεδίο επί της αρχικής εικόνας στο οποίο είναι ευαίσθητη κάθε μονάδα μεγαλώνει αλλά αναπαρίσταται πιο αδρά, επιτρέποντας έτσι υψηλότερου ιεραρχικού επιπέδου αναπαραστάσεις.

Χρησιμοποιώντας τον ίδιο συμβολισμό [13] με τα απλά πολυεπίπεδα νευρωνικά δίκτυα περιγράφουμε την πρόσθια τροφοδότηση ενός πολυεπίπεδου συνελικτικού δικτύου με εικόνες-όγκους τριών διαστάσεων, όπου η τρίτη διάσταση αντιπροσωπεύει τα κανάλια της εικόνας. Συγκεκριμένα, θεωρούμε εικόνα εισόδου στο επίπεδο l: $\mathbf{a}^{[l-1]} \in \mathbb{R}^{n_h^{[l-1]} \times n_w^{[l-1]} \times n_c^{[l-1]}}$ που έχει προκύψει ως έξοδος του προηγούμενου επιπέδου. Το επίπεδο l αποτελείται από $n_c^{[l]}$ το πλήθος 3Δ φίλτρα-χάρτες με βάρη $w_{ijk}^{[l],c}$ και πολώσεις $b^{[l],c}$, $c = 1, ..., n_c^{[l]}$, με διαστάσεις $[f_{conv}^{[l]}, f_{conv}^{[l]}, n_c^{[l-1]}]$. Το αποτέλεσμα της συνέλιξης του όγκου εισόδου με το όγκο-φίλτρο c, είναι μια εικόνα δύο διαστάσεων με pixel:

$$[z_c^{[l]}]_{(x,y)} = b^{[l],c} + \sum_i \sum_j \sum_k w_{ijk}^{[l],c} a_{x+i-1,y+i-1,k}^{[l-1]} \tag{2.29}$$

Συνηθίζεται, οι μετατοπίσεις των φίλτρων πάνω στην είσοδο (ή της εισόδου πάνω στο

φίλτρο) για τον υπολογισμό της συνέλιξης, να είναι μεγαλύτερες της μετατόπισης κατά ένα. Αυτό γίνεται για την μείωση του υπολογιστικού χρόνου που χρειάζεται η πράξη της συνέλιξης. Έτσι οι μετρητές στα αθροίσματα για i και j της (2.29) ανεβαίνουν κατά $s^{[l]}_{conv}$, που ονομάζεται stride.

Επίσης, για τον υπολογισμό των pixel στα άκρα της εικόνας, προστίθενται επιπλέον pixel στο περίγραμμα αυτής. Συνήθως, τα pixel αυτά έχουν την τιμή του πιο κοντινού πραγματικού pixel της εικόνας ή τη τιμή 0. Η διαδικασία αυτή ονομάζεται padding και η υπερπαράμετρος $p^{[l]}_{conv}$ καθορίζει πόσες σειρές και στήλες από pixel θα προστεθούν.

Το αποτέλεσμα της (2.29) είναι $n^{[l]}_c$ 2Δ εικόνες που μπορούν να συγχωνευθούν σε ένα όγκο με διαστάσεις $[n^{[l]}_{h,conv}, n^{[l]}_{w,conv}, n^{[l]}_c]$:

$$\begin{aligned} n^{[l]}_{h,conv} =& \lceil \frac{n^{[l-1]}_h + 2p^{[l]}_{conv} - f^{[l]}_{conv}}{s^{[l]}_{conv}} + 1 \rceil \\ n^{[l]}_{w,conv} =& \lceil \frac{n^{[l-1]}_w + 2p^{[l]}_{conv} - f^{[l]}_{conv}}{s^{[l]}_{conv}} + 1 \rceil \end{aligned} \tag{2.30}$$

Ο όγκος αυτός περνάει από μη γραμμική συνάρτηση ενεργοποίησης και εισέρχεται ως είσοδος στο επίπεδο υποδειγματοληψίας(pooling). Το επίπεδο αυτό αποτελείται από 2Δ φίλτρα μέσου όρου ή μεγίστου, με διαστάσεις: $[f^{[l]}_{conv}, f^{[l]}_{conv}]$, και $p^{[l]}_{pool}$, $s^{[l]}_{pool}$ που επιδρούν σε κάθε κανάλι ξεχωριστά.

Τελικά, προκύπτει εικόνα $\mathbf{a}^{[l]} \in \mathbb{R}^{n^{[l]}_h \times n^{[l]}_w \times n^{[l]}_c}$ με :

$$\begin{aligned} n^{[l]}_h =& \lceil \frac{n^{[l]}_{h,conv} + 2p^{[l]}_{pool} - f^{[l]}_{pool}}{s^{[l]}_{pool}} + 1 \rceil \\ n^{[l]}_w =& \lceil \frac{n^{[l]}_{w,conv} + 2p^{[l]}_{pool} - f^{[l]}_{pool}}{s^{[l]}_{pool}} + 1 \rceil \end{aligned} \tag{2.31}$$

όπου $\lceil \cdot \rceil$ η στρογγυλοποίηση προς τα πάνω στο κοντινότερο ακέραιο.

Κεφάλαιο 3

Ενισχυτική μάθηση

Η ενισχυτικής μάθησης συναντάται σε διάφορους κλάδους επιστημών όπως τον αυτόματο έλεγχο, την οικονομία κ.α. Στο πυρήνα της ασχολείται με το πρόβλημα της επιλογής δράσεων προς την επίτευξη στόχων. Στο κεφάλαιο αυτό, παρουσιάζονται βασικές ιδέες της κλασικής (tabular) θεώρησης της ενισχυτικής μάθησης στο πλαίσιο απλών περιβαλλόντων.

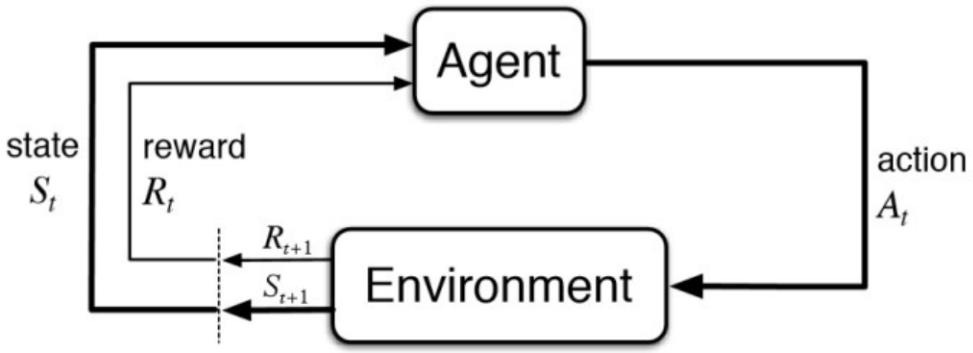
3.1 Η αλληλεπίδραση πράκτορα-περιβάλλοντος

Η τυπική μοντελοποίηση της αλληλεπίδρασης μεταξύ πράκτορα - περιβάλλοντος φαίνεται στο σχήμα 3.1. Ο πράκτορας λαμβάνει μια αναπαράσταση της κατάστασης του περιβάλλοντος και δρα, επηρρεάζοντας της επόμενη κατάσταση του περιβάλλοντος και λαμβάνοντας μια ανταμοιβή. Συγκεκριμένα κάθε διακριτή χρονική στιγμή t ο πράκτορας καλείται να πάρει μια απόφαση $A_t \in \mathcal{A}(s)$, με βάση την κατάσταση του περιβάλλοντος $S_t \in \mathcal{S}$. Η δράση αυτή έχει ως αποτέλεσμα την αλλαγή κατάστασης του περιβάλλοντος S_{t+1} και την επιστροφή ενός σήματος $R_{t+1} \in \mathbb{R}$, βάσει των οποίων ο πράκτορας επιλέγει τη επόμενη δράση του.

Το σήμα ανταμοιβής

Το σήμα ανταμοιβής είναι μια ακολουθία πραγματικών αριθμών που χρησιμοποιείται από τον πράκτορα για την λήψη αποφάσεων. Σε γενικές γραμμές, στόχος του πράκτορα είναι η μεγιστοποίηση του αθροίσματος των συνολικών ανταμοιβών που λαμβάνει από το περιβάλλον στο διηνεκές και όχι η μεγιστοποίηση της άμεσης ανταμοιβής. Η ιδέα αυτή εκφράζεται από την υπόθεση ανταμοιβής, σύμφωνα με την οποία, οποιοσδήποτε στόχος μπορεί να μοντελοποιηθεί ως η μεγιστοποίηση της προσδοκώμενης τιμής του αθροίσματος ενός βαθμωτού σήματος ανταμοιβών.

Εκ πρώτης όψεως, η ιδέα ότι ένα βαθμωτό σήμα είναι ικανό να περιγράψει τον στόχο ενός πράκτορα σε οποιοδήποτε περιβάλλον μπορεί να φανεί περιοριστική. Παρόλα αυτά, είναι πολύ ευέλικτη σε πλήθος πρακτικών εφαρμογών. Για παράδειγμα, σε ένα περιβάλλον όπως το σκάκι, μπορεί να δίνεται ανταμοιβή $+1$ στο τέλος μιας παρτίδας σε περίπτωση νίκης, -1 σε περίπτωση ήττας και 0 σε περίπτωση ισοπαλίας καθώς και σε όλες τις ενδιάμεσες χρονικές στιγμές. Σε ένα περιβάλλον όπου ένα ρομπότ καλείται να βρει το ελάχιστο μονοπάτι μεταξύ δύο σημείων,



Σχήμα 3.1: Αλληλεπίδραση πράκτορα-περιβάλλοντος (Πηγή: [21])

δίνεται ανταμοιβή -1 για κάθε βήμα του ρομπότ στο χώρο, και ο στόχος του πράκτορα είναι να φτάσει στο σημείο-στόχο με όσο το δυνατόν μικρότερες απώλειες. Για ένα ρομπότ που καλείται να ακολουθήσει μια συγκεκριμένη τροχιά στο χώρο των βαθμών ελευθερίας του, δίνεται ανταμοιβή $+1$ για κάθε χρονική στιγμή που ακολουθεί την επιθυμητή τροχιά.

Μαρκοβιανή Κατάσταση, κατάσταση περιβάλλοντος και αναπαράσταση αυτής από τον πράκτορα

Πολλές φορές η εσωτερική κατάσταση του περιβάλλοντος S^e δεν είναι προσβάσιμη από τον πράκτορα. Ένα περιβάλλον στο οποίο ο πράκτορας έχει πρόσβαση στην κατάσταση S^e ονομάζεται πλήρως παρατηρήσιμο. Σε αντίθετη περίπτωση, ο πράκτορας πρέπει να διατηρεί τη δική του αναπαράσταση της κατάστασης του περιβάλλοντος και λέμε ότι δρα σε ένα μερικώς παρατηρήσιμο περιβάλλον.

Σε ένα μερικώς παρατηρήσιμο περιβάλλον, ο πράκτορας καλείται να πάρει αποφάσεις χρησιμοποιώντας παρατηρήσεις O_t του περιβάλλοντος. Η συνεχής αλληλεπίδραση με το περιβάλλον παράγει το ιστορικό H_t για τη χρονική στιγμή t: $H_t = \{A_1, O_1, R_1, \ldots, A_t, O_t, R_t\}$. Ένας πράκτορας μπορεί να χρησιμοποιήσει το ιστορικό, προκειμένου να κατασκευάσει μια δική του αναπαράσταση S^a της κατάστασης του περιβάλλοντος S^e : $S^a_t = f(H_t)$. Ιδανικά θα θέλαμε το σήμα κατάστασης S^a_t να συνοψίζει πλήρως όλες τις πληροφορίες που είναι απαραίτητες για τον πράκτορα για την λήψη μιας απόφασης σε μια συμπαγή μορφή. Μια τέτοια κατάσταση, η οποία διατηρεί όλη την πληροφορία που περιλαμβάνεται στο ιστορικο ονομάζεται Μαρκοβιανή.

Θεωρώντας πεπερασμένο σύνολο καταστάσεων, μια κατάσταση S_t ονομάζεται Μαρκοβιανή ή λέγεται ότι ικανοποιεί την Μαρκοβιανή ιδιότητα αν και μόνο αν η επόμενη κατάσταση του περιβάλλοντος S_{t+1} εξαρτάται μόνο από την S_t :

$$Pr[S_{t+1}|S_t] = Pr[S_{t+1}|S_0, S_1, S_2, ..., S_t] \tag{3.1}$$

Ένα περιβάλλον, για το οποίο ισχύει η (3.1) για όλες τις καταστάσεις στο σύνολο καταστάσεων ονομάζεται Μαρκοβιανό. Ένα παράδειγμα των παραπάνω είναι ένας πράκτορας που καλείται να παίξει ένα βιντεοπαιχνίδι χρησιμοποιώντας τις εικόνες που εμφανίζονται στην

οθόνη. Ο πράκτορας δεν έχει πρόσβαση στην κατάσταση του παιχνιδιού S^e, η οποία βρίσκεται στην μνήμη του υπολογιστή. Η μόνη πληροφορία που λαμβάνει από το περιβάλλον είναι η εικόνα O_t. Ένας τρόπος να χρησιμοποιήσει αυτή την πληροφορία είναι να πάρει όλο το ιστορικό των διαδοχικών frames και να ορίσει την αναπαράσταση του ως $S_t^a = H_t$. Η χρήση όλου του ιστορικού ως κατάστασης ικανοποιεί εξ ορισμού την Μαρκοβιανή ιδιότητα, αλλά δεν είναι ιδιαίτερα χρήσιμη καθώς εμπεριέχει πλεονάζουσα πληροφορία. Στην συνέχεια, όπου αναφερόμαστε στην κατάσταση θεωρούμε την αναπαράσταση S^a του πράκτορα

3.2 Μαρκοβιανές διαδικασίες απόφασης

Οι Μαρκοβιανές διαδικασίες απόφασης (ΜΔΑ) αποτελούν το θεωρητικό μοντέλο αναπαράστασης της αλληλεπίδρασης μεταξύ πράκτορα-περιβάλλοντος. Θεωρώντας σύνολα διακριτών καταστάσεων $\mathcal{S}$ και δράσεων $\mathcal{A}$ με πεπερασμένο πλήθος στοιχείων, η ΜΔΑ περιγράφεται πλήρως από την τριάδα $M = (\mathcal{S}, \mathcal{A}, p)$, όπου η κατανομή πιθανότητας p περιγράφει τη δυναμική του περιβάλλοντος και ορίζεται ως:

$$p(s', r|s, a) = Pr[S_{t+1} = s', R_{t+1} = r|S_t = s, A_t = a] \tag{3.2}$$

Δεδομένης της $p(s', r|s, a)$, μπορούμε να υπολογίσουμε την πιθανότητα μετάβασης κατάστασης:

$$p(s'|s, a) = P_{s,s'}^a = Pr[S_{t+1} = s'|S_t = s, A_t = a] = \sum_{r \in \mathcal{R}} p(s', r|s, a) \tag{3.3}$$

καθώς και την προσδοκώμενη ανταμοιβή για ζεύγη κατάστασης-δράσης:

$$r(s, a) = R_s^a = \mathbb{E}[R_{t+1}|S_t = s, A_t = a] = \sum_r r \sum_{s' \in \mathcal{S}} p(s', r|s, a) \tag{3.4}$$

Επιστροφή

Απώτερος στόχος ενός πράκτορα είναι η μεγιστοποίηση της προσδοκώμενης επιστροφής. Η επιστροφή τη χρονική στιγμή t είναι μια τυχαία μεταβλητή που ορίζεται ως το άθροισμα όλων των μελλοντικών άμεσων ανταμοιβών:

$$G_t = R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + ... + \gamma^{T-1} R_T \tag{3.5}$$

όπου γ ονομάζεται παράγοντας έκπτωσης και $T \in \mathbb{N} \cup \{\infty\}$.

Επεισοδιακές και συνεχιζόμενες ΜΔΑ

Σε κάποιες εφαρμογές υπάρχει μια φυσική έννοια του τελικού βήματος T, δηλαδή η αλληλεπίδραση μεταξύ πράκτορα και περιβάλλοντος διαιρείται σε ακολουθίες από εμπειρίες που ονομάζονται επεισόδια. Σε αυτές τις περιπτώσεις, $T \in \mathbb{N}$, ο παράγοντας έκπτωσης $0 \leq \gamma \leq 1$

και η επιστροφή ονομάζεται επεισοδιακή. Θεωρούμε ότι το σύνολο καταστάσεων στις επεισοδιακές ΜΔΑ περιλαμβάνει μια ειδική κατάσταση που ονομάζεται τερματική και ακολουθείται από την εναρκτήρια κατάσταση του επόμενου επεισοδίου. Παραδείγματα τέτοιων ΜΔΑ, είναι ένα επιτραπέζιο παιχνίδι, όπου η χρονική στιγμή T σηματοδοτεί το τέλος μιας παρτίδας, ή ένα βιντεοπαιχνίδι, όπου η χρονική στιγμή T σηματοδοτεί το game over.

Αντίθετα υπάρχουν εφαρμογές, στις οποίες οι πράκτορες αλληλεπιδρούν με το περιβάλλον επ' άπειρον και ονομάζονται συνεχιζόμενες. Σε αυτή την περίπτωση, θεωρούμε $T \to \infty$, και $0 \leq \gamma < 1$. Δεδομένου ότι ο στόχος ενός πράκτορα είναι η επιλογή δράσεων προς τη μεγιστοποίηση της μελλοντικής επιστροφής, η τιμή $\gamma = 1$ στο πλαισιο μιας συνεχιζόμενης εργασίας, θα καθιστούσε αδύνατη την σύγκριση μεταξύ διαφορετικών τιμών της τυχαίας μεταβλητής G_t. Παρόλα αυτά για $0 \leq \gamma < 1$ και φραγμένη ακολουθία $\{R_t\}$ το άπειρο άθροισμα G_t είναι πεπερασμένο. Παραδείγματα συνεχιζόμενων ΜΔΑ είναι ένα ρομπότ με μεγάλο χρόνο ζωής και γενικά πληθώρα προβλημάτων αυτομάτου ελέγχου.

Παράγοντας έκπτωσης

Σε κάθε περίπτωση, ο παράγοντας έκπτωσης γ καθορίζει την αξία των μελλοντικών ανταμοιβών. Μία ανταμοιβή τη χρονική στιγμή $t + k$ συμβάλλει $\gamma^k R_{t+k+1}$ στο άθροισμα της επιστροφής. Συνεπώς, ο παράγοντας έκπτωσης ρυθμίζει το πόσο σημαντικά είναι οι μακροπρόθεσμες ανταμοιβές για τον πράκτορα. Για $\gamma = 0$, η διαδικασία μεγιστοποίησης της προσδοκώμενης επιστροφής ανάγεται στην επιλογή της δράσης με τη μεγαλύτερη άμεση ανταμοιβή. Για $\gamma \to 1$, ο πράκτορας δίνει περισσότερη αξία στις μακροπρόθεσμες ανταμοιβές.

Πολιτική

Ο τρόπος λήψης των αποφάσεων από τον πράκτορα καθορίζεται από την πολιτική που ακολουθεί. Η πολιτική ορίζεται ως μια συνάρτηση $\pi : S \mapsto p(A)$, που αντιστοιχεί καταστάσεις σε κατανομές πιθανότητας στο χώρο δράσης και θεωρούμε ότι είναι stationary:

$$\pi(a|s) = Pr[A_t = a|S_t = s] \tag{3.6}$$

3.2.1 Συνάρτηση αξίας

Η συνάρτηση αξίας κατάστασης αποτελεί κεντρική ιδέα των ΜΔΑ και ορίζεται ως η συνάρτηση $v_\pi : \mathcal{S} \mapsto \mathbb{R}$ που δίνει τη προσδοκώμενη επιστροφή από μια κατάσταση s, υποθέτοντας ότι ο πράκτορας επιλέγει δράσεις βάσει μιας πολιτικής π :

$$v_\pi(s) = \mathbb{E}_\pi[G_t|S_t = s],\ s \in \mathcal{S} \tag{3.7}$$

Αντίστοιχα μπορούμε να ορίσουμε την συνάρτηση αξίας κατάστασης - δράσης $q_\pi : \mathcal{S} \times \mathcal{A} \mapsto \mathbb{R}$ που δίνει την προσδοκώμενη επιστροφή από μια κατάσταση s, υποθέτοντας ότι ο πράκτορας επιλέγει τη δράση a και στη συνέχεια συμπεριφέρεται βάσει της πολιτικής π:

$$q_\pi(s, a) = \mathbb{E}_\pi[G_t|S_t = s, A_t = a],\ s \in \mathcal{S}, a \in \mathcal{A} \tag{3.8}$$

Στα πλαίσια των επεισοδιακών ΜΔΑ, θεωρούμε ότι για την τελική κατάσταση S_T ισχύει $v_\pi(S_T) = 0$ και $q_\pi(S_T, a) = 0$ για κάθε $a \in \mathcal{A}$.

3.2.2 Εξίσωση Bellman

Μια βασική ιδιότητα των συναρτήσεων αξίας είναι ότι μπορούν να εκφραστούν αναδρομικά χρησιμοποιώντας την παρατήρηση ότι:

$$G_t = R_{t+1} + \gamma G_{t+1} \tag{3.9}$$

και το νόμο της συνολικής προσδοκίας $\mathbb{E}[X] = \mathbb{E}[\mathbb{E}[X|Y]]$ παίρνουμε:

$$v_\pi(s) = \mathbb{E}_\pi[R_{t+1} + \gamma G_{t+1}|S_t = s] = \mathbb{E}_\pi[R_{t+1} + \gamma v_\pi(s')|S_t = s] \tag{3.10}$$

και αντίστοιχα για την συνάρτηση αξίας κατάστασης-δράσης:

$$q_\pi(s, a) = \mathbb{E}_\pi[R_{t+1} + \gamma q_\pi(s', a')|S_t = s, A_t = a] \tag{3.11}$$

Αναπτύσσοντας την (3.10) για τις δυνατές δράσεις από την κατάσταση s σύμφωνα με την πολιτική π και για τη δυναμική της ΜΔΑ έχουμε:

$$v_\pi(s) = \sum_{a\in\mathcal{A}} \pi(a|s) \sum_r \sum_{s'\in\mathcal{S}} p(r, s'|s, a)[r + \gamma v_\pi(s')] \tag{3.12}$$

η οποία αποτελεί την εξίσωση Bellman για την συνάρτηση αξίας κατάστασης. Χρησιμοποιώντας τις (3.3), (3.4), η εξίσωση Bellman μπορεί να γραφεί με την βοήθεια των R^a_s, $P^a_{s,s'}$:

$$v_\pi(s) = \sum_{a\in\mathcal{A}} \pi(a|s)[R^a_s + \gamma \sum_{s'\in\mathcal{S}} P^a_{s,s'} v_\pi(s')] \tag{3.13}$$

Θεωρώντας ότι η απόφαση για δραση a στην κατάσταση s έχει ληφθεί, αναπτύσσουμε την (3.11) για τις δυνατές δράσεις a' από την κατάσταση s' σύμφωνα με την πολιτική π και τη δυναμική της ΜΔΑ έχουμε:

$$q_\pi(s, a) = \sum_{a'\in\mathcal{A}} \sum_r \sum_{s'\in\mathcal{S}} \pi(a|s) p(r, s'|s, a)[r + \gamma q_\pi(s', a')] \tag{3.14}$$

η οποία αποτελεί την εξίσωση Bellman για την συνάρτηση αξίας κατάστασης - δράσης. Χρησιμοποιώντας τις (3.3), (3.4), η εξίσωση Bellman μπορεί να γραφεί με την βοήθεια των R^a_s, $P^a_{s,s'}$

$$q_\pi(s, a) = R^a_s + \gamma \sum_{s'\in\mathcal{S}} P^a_{s,s'} \sum_{a'\in\mathcal{A}} \pi(a'|s') q_\pi(s', a') \tag{3.15}$$

Οι εξισώσεις Bellman εκφράζουν την συνάρτηση αξίας $v_\pi(s)$ και $q_\pi(s, a)$ βάσει των συναρτήσεων αξίας στην επόμενη κατάσταση ή ζεύγος κατάστασης-δρασης αντίστοιχα. Μπορούμε

να υπολογίσουμε την αξία μιας κατάστασης s ως την σταθμισμένη σύμφωνα με την πολιτική π μέση αξία κάθε ζεύγους (s, a):

$$v_\pi(s) = \sum_{a \in \mathcal{A}} \pi(a|s) q_\pi(s, a) \tag{3.16}$$

Αντίστοιχα η αξία ενός ζεύγους κατάσταση-δράσης ισούται με το άθροισμα της άμεσης ανταμοιβής που δίνει το περιβάλλον και την εκπτωτούμενη, σταθμισμένη σύμφωνα με την δυναμική του περιβάλλοντος, μέση αξία κάθε πιθανής επόμενης κατάστασης s':

$$q_\pi(s, a) = R_s^a + \gamma \sum_{s' \in \mathcal{S}} P_{s,s'}^a v_\pi(s') \tag{3.17}$$

3.3 Δυναμικός Προγραμματισμός

Ο δυναμικός προγραμματισμός αποτελεί μια ευρέως διαδεδομένη οικογένεια μεθόδων επίλυσης μιας κατηγορίας προβλημάτων, που διέπονται από βέλτιστη-υποδομή(optimal substructure). Τα προβλήματα αυτά, χαρακτηρίζονται από την ιδιότητα ότι μπορούν επιλυθούν βέλτιστα, με αναδρομική σύνθεση των λύσεων μικρότερων υπο-προβλημάτων(optimal substructure). Στα πλαίσια της ενισχυτικής μάθησης, οι συναρτήσεις αξίας παρουσιάζουν βέλτιστη υποδομή.

3.3.1 Αξιολόγηση πολιτικής

Η επίλυση των (3.13), (3.15) ως προς τις συναρτήσεις αξίας είναι απαραίτητη για την αξιολόγηση μιας πολιτικής. Ορίζουμε τις ανταμοιβές που προκύπτουν από την λήψη αποφάσεων κάτω από την πολιτική π, καθώς και την δυναμική της ΜΔΑ για δράσεις σύμφωνα με μια πολιτική π :

$$R_s^\pi = \sum_a \pi(a|s) R_s^a \tag{3.18}$$

$$P_{s,s'}^\pi = \sum_a \pi(a|s) P_{s,s'}^a \tag{3.19}$$

Θεωρώντας διανύσματα $\mathbf{v}^\pi \in \mathbb{R}^{|\mathcal{S}|\times 1}$, $\mathbf{R}^\pi \in \mathbb{R}^{|\mathcal{S}|\times 1}$ με στοιχεία R_s^π από την (3.18) και πίνακα $\mathbf{P}^\pi \in \mathbb{R}^{|\mathcal{S}|\times|\mathcal{S}|}$ με στοιχεία $P_{s,s'}^\pi$ από την (3.19), η (3.13) μπορεί να γραφεί σε μητρική μορφή:

$$\mathbf{v}^\pi = \mathbf{R}^\pi + \gamma \mathbf{P}^\pi \mathbf{v}^\pi \tag{3.20}$$

η οποία αποτελεί γραμμική εξισωση με αγνώστους τις συναρτήσεις αξίας για κάθε κατάσταση και μπορεί να λυθεί με πληθώρα γνωστών μεθόδων.

Εναλλακτικά [24], μπορούμε να ορίσουμε τελεστή $\mathcal{T}^\pi$ για stationary πολιτική π ως ένα mapping $\mathcal{T} : \Omega(S) \mapsto \Omega(S)$, όπου Ω ο χώρος των φραγμένων συναρτήσεων v:

$$(\mathcal{T}^\pi v)(s) = R_s^\pi + \gamma \sum_{s'} P_{s,s'}^\pi v(s') \tag{3.21}$$

όπου R_s^π και $P_{s,s'}^\pi$ δίνονται από τις (3.18), (3.19) αντίστοιχα και η (3.13) γράφεται:

$$(\mathcal{T}^\pi v_\pi)(s) = v_\pi(s) \tag{3.22}$$

Αποδεικνύεται ότι [24], [2], για $0 < \gamma < 1$, η (3.22) έχει μοναδική λύση v_π η οποία μπορεί να βρεθεί από την επαναληπτική εφαρμογή του τελεστή $\mathcal{T}^\pi$ για οποιαδήποτε αρχική τιμή v_0. Η ακολουθία συγκλίνει στη v_π με γεωμετρικό ρυθμό.

Το αποτέλεσμα αυτό αποτελεί την βάση του αλγορίθμου αξιολόγησης πολιτικής. Δεδομένης πολιτικής π και αρχικοποίησης v_0 υπολογίζουμε την αξία κατάστασης v_π επαναληπτικά με εφαρμογή του Bellman τελεστή (3.21). Σε διανυσματική μορφή:

Algorithm 1: Αξιολόγηση πολιτικής

Input: π, **Output**: $\mathbf{v}_\pi$

Initialize $\mathbf{v}_0$ randomly(say 0 for every state)

for $k \leftarrow 0, 1, \ldots$ **do**

  $\mathbf{v}_{k+1} = \mathbf{R}^\pi + \gamma \mathbf{P}^\pi \mathbf{v}_k$

Αντίστοιχα αποτελέσματα μπορούν να εξαχθούν και για την συνάρτηση αξίας κατάστασης-δράσης (3.15)

3.3.2 Βέλτιστη εξίσωση Bellman

Στόχος ενός πράκτορα σε μια ΜΔΑ είναι εύρεση της βέλτιστης πολιτικής π^*. Η βέλτιστη πολιτική π^* ορίζεται ως μια πολιτική που επιφέρει την μεγαλύτερη δυνατή προσδοκώμενη επιστροφή από οποιαδήποτε κατάσταση. Ορίζουμε τη βέλτιστη συνάρτηση αξίας κατάστασης v^* και τη βέλτιστη συνάρτηση αξίας κατάστασης - δράσης q^* ως:

$$v^*(s) = \max_\pi v_\pi(s),\ \forall s \in \mathcal{S} \tag{3.23}$$

$$q^*(s,a) = \max_\pi q_\pi(s,a),\ \forall (s,a) \in \mathcal{S} \times \mathcal{A} \tag{3.24}$$

Μια πολιτική π^* ονομάζεται βέλτιστη αν:

$$v_{\pi^*}(s) = v^*(s) \forall s \in \mathcal{S} \tag{3.25}$$

Η (3.24) δίνει την προσδοκώμενη επιστροφή αν στην κατάσταση s επιλέξουμε την δράση a και στη συνέχεια ακολουθήσουμε μια βέλτιστη πολιτική π^*. Συνεπώς μπορούμε να εκφράσουμε την $q^*(s,a)$ με αναλογο τρόπο με την (3.17):

$$q^*(s,a) = R_s^a + \gamma \sum_{s' \in \mathcal{S}} P_{s,s'}^a v^*(s') \tag{3.26}$$

Αναφορικά με την βέλτιστη συνάρτηση αξίας $v^*(s)$ σε μια κατάσταση s, θεωρούμε ότι ισούται με την προσδοκώμενη επιστροφή για την καλύτερη δυνατή δράση από την κατάσταση s. Αποδεικνύεται [2]:

$$v^*(s) = \max_a q^*(s,a) \tag{3.27}$$

Από τις (3.26), (3.27) εξάγουμε τις βέλτιστες εξισώσεις Bellman για τις βέλτιστες συναρτησεις αξίας v^*, q^*:

$$v^*(s) = \max_a [R_s^a + \gamma \sum_{s' \in \mathcal{S}} P_{s,s'}^a v^*(s')] \tag{3.28}$$

$$q^*(s,a) = R_s^a + \gamma \sum_{s' \in \mathcal{S}} P_{s,s'}^a \max_a q^*(s,a) \tag{3.29}$$

Οι εξισώσεις (3.28) (ή (3.29)) σχηματίζουν ένα σύστημα μη γραμμικών εξισώσεων με αγνώστους v^* (ή q^*). Όπως θα δούμε παρακάτω, η επίλυση του συστήματος, αρκεί για την εύρεση μιας βέλτιστης πολιτικής.

3.3.3 Επανάληψη αξίας

Ορίζουμε τελεστή $\mathcal{T}^*$ ως ένα mapping $\mathcal{T}^* : \Omega(S) \mapsto \Omega(S)$, όπου Ω ο χώρος των φραγμένων συναρτήσεων v:

$$(\mathcal{T}^* v)(s) = \max_a [R_s^a + \gamma \sum_{s' \in \mathcal{S}} P_{s,s'}^a v(s')] \tag{3.30}$$

Με τη βοήθεια του τελεστή $\mathcal{T}^*$, η (3.28) γράφεται:

$$(\mathcal{T}^* v^*)(s) = v^*(s) \tag{3.31}$$

Αποδεικνύεται ότι [24], [2] για $0 < \gamma < 1$, η (3.31) έχει μοναδική λύση v^* η οποία μπορεί να βρεθεί από την επαναληπτική εφαρμογή του τελεστή $\mathcal{T}^*$ για οποιαδήποτε αρχική τιμή v_0. Η ακολουθία συγκλίνει στη v^* με γεωμετρικό ρυθμό.

Συνεπώς, για κάθε πεπερασμένη ΜΔΑ με $0 < \gamma < 1$ υπάρχει μοναδική $v^*(s) = \max_\pi v_\pi(s)$ για κάθε $s \in \mathcal{S}$ και μοναδική $q^*(s,a) = \max_\pi q_\pi(s,a)$ για κάθε $(s,a) \in \mathcal{S} \times \mathcal{A}$. Ακόμη, αποδεικνύεται [24] ότι μια πολιτική που δρα άπληστα ως προς την v^*(ή q^* μέσω της (3.26)) είναι βέλτιστη, δηλαδή ικανοποιεί την (3.25). Άρα έχουμε πλέον την δυνατότητα να βρούμε την βέλτιστη πολιτική για μια ΜΔΑ.

Το αποτέλεσμα αυτό αποτελεί την βάση του αλγορίθμου επανάληψης αξίας. Δεδομένης μιας ΜΔΑ και αρχικοποίησης v_0^* υπολογίζουμε την βέλτιστη συνάρτηση αξίας κατάστασης v^* επαναληπτικά με εφαρμογή του βέλτιστου Bellman τελεστή (3.31). Τέλος, για την εύρεση της βέλτιστης πολιτικής, δρούμε άπληστα ως προς την συνάρτηση v^*:

$$\pi^*(a|s) = \begin{cases} 1, \text{ αν } a = \underset{a'}{\operatorname{argmax}} [R_s^{a'} + \gamma \sum_{s'} P_{s,s'}^{a'} v^*(s')] \\ 0, \text{ αλλιώς} \end{cases} \tag{3.32}$$

Αντίστοιχα αποτελέσματα μπορούν να εξαχθούν και για την βέλτιστη συνάρτηση αξίας κατάστασης-δράσης (3.26). Μάλιστα, θα δούμε ότι η χρήση της q^* καθιστά δυνατή τη χρήση των παραπάνω ιδεων στη περίπτωση που η δυναμική του περιβάλλοντος δεν είναι γνωστή.

Algorithm 2: Επανάληψη αξίας

Output: $\mathbf{v}^*_\pi$, π
Initialize $\mathbf{v}^*_0$ randomly(say 0 for every state)
for $k \leftarrow 0, 1, \ldots$ **do**
  $\mathbf{v}^*_{k+1} = \max_a[\mathbf{R}^a + \gamma \mathbf{P}^a \mathbf{v}^*_k]$
$\pi^* =$ greedy w.r.t. $\mathbf{v}^*$

3.3.4 Βελτίωση πολιτικής

Στον αλγόριθμο επανάληψης τιμής είδαμε ότι μια πολιτική που δρα άπληστα ως προς την βέλτιστη συνάρτηση αξίας, είναι βέλτιστη. Γενικά αποδεικνύεται ότι μια πολιτική π που δρα άπληστα ως προς την συνάρτηση αξίας v_{π_0} μιας πολιτικής π_0, έχει συνάρτηση αξίας v_π για την οποία ισχύει ότι $v_\pi(s) \geq v_{\pi_0}$ για κάθε $s \in \mathcal{S}$.

Πιο αναλυτικά, θεωρούμε stationary πολιτική π με συνάρτηση αξίας v_π και μια πολιτική $\pi^{'}$ η οποία δρα άπληστα ως προς την v_π, δηλαδή $\pi' \leftarrow greedy(v_\pi)$. Τότε αποδεικνύεται [22], [24], [2] ότι $v_{\pi'}(s) \geq v_\pi$ για κάθε $s \in \mathcal{S}$ δηλαδή η $\pi^{'}$ είναι καλύτερη από την π. Στη περίπτωση ισότητας, η συνάρτηση $v_{\pi'}$ είναι βέλτιστη: $v_{\pi'} = v^*$ και η πολιτική π' είναι βέλτιστη.

Το παραπάνω θεώρημα είναι γνωστό ως θεώρημα βελτίωσης πολιτικής και αποτελεί τη βάση του αλγορίθμου Βελτίωσης πολιτικής. Ξεκινώντας από μια τυχαία αρχικοποιημένη πολιτική π, υπολογίζουμε την συνάρτηση αξίας της v_π με χρήση του αλγορίθμου αξιολόγησης πολιτικής και ορίζουμε την πολιτική π' που επιλέγει δράσεις άπληστα σύμφωνα με την v_π και επαναλαμβάνουμε την διαδικασία με την καινούργια πολιτική. Η καινούργια πολιτική θα αποτελεί βελτίωση της προηγούμενης και αποδεικνύεται ότι για πεπερασμένες ΜΔΑ η διαδικασία τερματίζει σε πεπερασμένο αριθμό βημάτων και η πολιτική που προκύπτει είναι βέλτιστη.

Algorithm 3: Επανάληψη πολιτικής

Output: $\mathbf{v}^*_\pi$, π^*
Initialize π_0 randomly(say uniform)
$\mathbf{v}_0 =$ policy_evaluation(π_0) [alg. 1]
repeat
  $\pi_{k+1} \leftarrow$ greedy w.r.t $\mathbf{v}_k$
  $\mathbf{v}_{k+1} =$ policy_evaluation(π_{k+1}) [alg. 1]
  $k \leftarrow k + 1$
until $\mathbf{v}_k = \mathbf{v}_{k-1}$;

3.3.5 Παραλλαγές επανάληψης τιμής

Ο αλγόριθμος επανάληψης τιμής όπως εκφράστηκε σε μητρική μορφή είναι ένας σύγχρονος αλγόριθμος στον οποίο διατηρούμε δύο αντίγραφα της συνάρτησης αξίας $\mathbf{v}^*_{old}$, $\mathbf{v}^*_{new}$. Παρόλα αυτά μπορούμε να κάνουμε ασύγχρονα in-place updates για όλες τις καταστάσεις $s \in \mathcal{S}$ με χρήση της (3.28), χρησιμοποιώντας κάθε φορά την νεότερη τιμή αξίας για μια κατάσταση. Ε-

φόσον, ανανεώνουμε συνεχώς τις τιμές αξίας για όλες τις καταστάσεις ο αλγόριθμος συγκλίνει [21].

Αντι να ανανεώνουμε τη συνάρτηση αξίας για τυχαίες καταστάσεις μπορούμε να θεωρήσουμε μια προτεραιότητα με βάση το το απόλυτο σφάλμα Bellman:

$$\epsilon(s) = |\max_a[R_s^a + \gamma \sum_{s'} P_{s,s'}^a v^*(s')] - v^*(s)| \tag{3.33}$$

και να δίνουμε προτεραιότητα στις καταστάσεις με το μεγαλύτερο σφάλμα.

Τέλος, αν έχουμε ένα πράκτορα που δειγματοληπτεί εμπειρίες $(S_t, A_t, R_{t+1}, S_{t+1})$ από το περιβάλλον, μπορούμε να ανανεώνουμε τη συνάρτηση αξίας μόνο για τις καταστάσεις που βλέπει ο πράκτορας.

3.3.6 Γενικευμένη βελτίωση πολιτικής(GPI)

Η βελτίωση πολιτικής συνίσταται από δύο αλληλένδετες διαδικασίες: τον υπολογισμός της συνάρτησης αξίας για μια πολιτική(εκτίμηση αξίας) και την χρήση μιας νέας πολιτικής άπληστης ως προς την συνάρτηση αυτή. Η ιδέα αυτή αποτελεί τη βάση της γενικευμένη βελτίωση πολιτικής [22], σύμφωνα με την οποία αφήνουμε τις δύο διαδικασίες να αλληλεπιδρούν σειριακά ανεξαρτήτως δεδομένων του προβλήματος και εσωτερικών λεπτομερειών των δύο διαδικασιών. Εφόσον, οι δύο διαδικασίες επιλέγουν και ανανεώνουν όλες τις καταστάσεις ή δράσεις, ο συνολικός αλγόριθμος συνήθως συγκλίνει.

Γενικά, αν και οι δύο διαδικασίες σταθεροποιηθούν δηλαδή η αποτίμηση της πολιτικής δεν αλλάζει σημαντικά την συνάρτηση αξίας και η βελτίωση πολιτικής δεν επηρεάζει την ήδη υπάρχουσα πολιτική, τότε η συνάρτηση αξίας και η πολιτική πρέπει να είναι βέλτιστες. Η συνάρτηση αξίας σταθεροποιείται όταν είναι συμβατή με την υπάρχουσα πολιτική και η βελτίωση πολιτικής όταν είναι άπληστη ως προς την συνάρτηση αξίας. Συνεπώς και οι δύο διαδικασίες σταθεροποιούνται όταν έχει βρεθεί μια πολιτική άπληστη ως προς την συνάρτηση αξίας της, γεγονός που υπονοεί την εξίσωση Bellman.

3.4 Ενισχυτική μάθηση για άγνωστο μοντέλο

Στο δυναμικό προγραμματισμό θεωρούμε ότι η δυναμική $p(s', r|s, a)$ της ΜΔΑ είναι γνωστή. Σε πολλά πρακτικά προβλήματα αυτό δεν ισχύει και ο πράκτορας καλείται να βρει βέλτιστες πολιτικές από εμπειρίες του με το περιβάλλον, δηλαδή τετράδες (S, A, R, S').

Η ιδέα που ακολουθείται είναι η γενικευμένη βελτίωση πολιτικής, δηλαδή ο συνεχής υπολογισμός της συνάρτησης αξίας μιας πολιτικής, ακολουθούμενος από την βελτίωση της πολιτικής. Παρατηρούμε ότι η εξίσωση Bellman (3.28), απαιτεί την γνώση της δυναμικής της ΜΔΑ και άρα δεν μπορεί να χρησιμοποιηθεί για την άπληστη βελτίωση της πολιτικής(εφαρμογή max). Αντίθετα, σύμφωνα με τη (3.27) μπορούμε να υλοποιήσουμε την βελτίωση πολιτικής δρώντας άπληστα ως προς την συνάρτηση αξίας κατάστασης-δράσης.

3.4.1 Αξιολόγηση Πολιτικής Monte Carlo

Στο πρόβλημα της πρόβλεψης πολιτικής, δοθείσης μιας πολιτικής π, καλούμαστε να υπολογίσουμε την συνάρτηση αξίας v_π ή q_π από εμπειρίες. Οι μέθοδοι Monte Carlo βασίζονται στην χρήση της μέσης τιμής δειγμάτων επιστροφών για το υπολογισμό της·συνάρτησης αξίας. Δεδομένου ότι η αξία μιας κατάστασης εκτιμάται από την (3.7) ως η προσδοκώμενη τιμή της τυχαίας μεταβλητής της επιστροφής G_t ευρισκόμενοι σε μια κατάσταση s, μπορούμε να την υπολογίσουμε ως την μέση τιμή πολλών realisation της τυχαίας μεταβλητής ξεκινώντας από μια κατάσταση s.

Συγκεκριμένα, θεωρούμε τον υπολογισμό της $v_\pi(s)$ δεδομένων επεισοδίων στα οποία ο πράκτορας ακολουθεί την πολιτική π και περνάει από την κατάσταση s. Κάθε πέρασμα από την κατάσταση s, ονομάζεται επίσκεψη στην s. Αν ο πράκτορας επισκέπτεται την κατάσταση s πολλές φορές, θεωρούμε την πρώτη επίσκεψή του σε αυτή. Ο αλγόριθμος Monte Carlo πρώτης επίσκεψης υπολογίζει την $v_\pi(s)$ ως το μέσο όρο των επιστροφών από την πρώτη επίσκεψη στην s. Ανάλογα ο αλγόριθμος Monte Carlo κάθε επίσκεψης χρησιμοποιεί το μέσο όρο επιστροφών από την s για κάθε επίσκεψη. Και στις δύο περιπτώσεις, ο αλγόριθμος συγκλίνει στην πραγματική $v_\pi(s)$, όσο ο αριθμός των επισκέψεων(ή των πρώτων επισκέψεων) στην s τείνει στο άπειρο $N(s) \to \infty$ [22]

Algorithm 4: Πρόβλεψη MC

Output: $\mathbf{V}_\pi$
Initialize π randomly(say uniform), $V_\pi(s) \forall s$
Returns(s) list of empty lists $\forall s \in \mathcal{S}$
repeat
  Generate an episode using π
  for *each state s appearing in the episode* **do**
    $G \leftarrow$ Return following first/every occurence of s
    Append G to Returns(s)
    $V(s) \leftarrow$ online average(Returns(s))
until *Forever*;

Ο υπολογισμός του μέσου όρου επιστροφών στην περίπτωση του Monte Carlo πρώτης επίσκεψης μπορεί να γίνει αυξητικά για κάθε κατάσταση, μετά το πέρας του επεισοδίου:

$$\mu_k = \mu_{k-1} + \frac{1}{k}(x_k - \mu_{k-1}) \tag{3.34}$$

Ενώ στην περίπτωση του every-visit Monte Carlo:

$$\begin{aligned} \Delta &= \mu_b - \mu_{k-1} \\ N_k &= N_{k-1} + N_b \\ \mu_k &= \mu_{k-1} + \Delta(\frac{N_b}{N_k}) \end{aligned} \tag{3.35}$$

όπου N_b ο αριθμός επισκέψεων της s στο νέο επεισόδιο, μ_b η μέση επιστροφή της κατάστασης s στο νέο επεισόδιο.

Βλέπουμε ότι, σε αντίθεση με το δυναμικό προγραμματισμό, ο υπολογισμός της συνάρτησης αξίας για μια κατάσταση γίνεται με βάση την ακολουθία καταστάσεων στο επεισόδιο που ακολουθήθηκε και όχι από όλες τις δυνατές επόμενες καταστάσεις. Ακόμη, λαμβάνουμε υπόψη όλες τις διαδοχικές καταστάσεις του επεισοδίου και όχι μόνο τις καταστάσεις για ένα βήμα. Οι διαφορές αυτές αποτελούν βασικά κριτήρια κατηγοριοποίησης των δύο αλγορίθμων. Είναι σημαντικό ότι, στο Monte Carlo οι εκτιμήσεις της συνάρτησης αξίας μιας κατάστασης είναι ανεξάρτητες από αυτές των υπόλοιπων καταστάσεων, σε αντιδιαστολή με το δυναμικό προγραμματισμό όπου σύμφωνα με την εξίσωση (3.13), η εκτίμησή για την αξία της κατάστασης s, βασίζεται στην εκτίμησή για την κατάσταση s'(bootstrap).

3.4.2 Αξιολόγηση Πολιτικής Χρονικών Διαφορών

Ένα από τα προβλήματα που παρουσιάζονται στην εκτίμηση Monte Carlo, είναι η υψηλή διακύμανση. Δεδομένου ότι η εκτίμηση της συνάρτησης αξίας γίνεται μέσω της προσδοκώμενης τιμής των επιστροφών, και ότι οι επιστροφές αποτελούν άθροισμα τυχαίων μεταβλητών, η διακύμανση αυξάνεται με το μήκος του επεισοδίου.

Οι μέθοδοι Χρονικών Διαφορών(Temporal Difference) συνδυάζουν ιδέες από το Monte Carlo και το δυναμικό προγραμματισμό, υπό την έννοια ότι, όπως το MC, εκτιμούν τη συνάρτηση αξίας για δράσεις που όντως έλαβαν χώρα και όπως στο δυναμικό προγραμματισμό, βασίζουν την εκτίμηση τους για μια κατάσταση s σε εκτιμήσεις επόμενων καταστάσεων(bootstrap).

Αξιολογηση με TD(0)

Συγκεκριμένα, στην μέθοδο TD η εκτίμηση της συνάρτησης αξίας για μια κατάσταση s, γίνεται μέσω update rule της μορφής:

$$New\_estimation \leftarrow old\_estimation + \alpha(target - old\_estimation) \tag{3.36}$$

το οποίο χρησιμοποιείται ως:

$$V(S_t) \leftarrow V(S_t) + \alpha(R_{t+1} + \gamma V(S_{t+1}) - V(S_t)) \tag{3.37}$$

Ο όρος $R_{t+1} + \gamma V(s_{t+1})$ αποτελεί το στόχο προς τον οποίο ενημερώνεται η εκτίμηση $V(S_t)$ και συνολικά όρος $R_{t+1} + \gamma V(S_{t+1}) - V(S_t)$ ονομάζεται td-error. Ο συντελεστής $\alpha \leq 1$ καθοριζει την ταχύτητα σύγκλισης και ονομάζεται συντελεστής βήματος.

Αποδεικνύεται [24] ότι αν οι ενημερώσεις συγκλίνουν, τότε πρέπει να συγκλίνουν σε συνάρτηση V, που μηδενίζει το μέσο td-error για όλες τις καταστάσεις που δειγματοληπτούνται από την πολιτική απείρως συχνά. Δηλαδή: $\mathbb{E}[R_{t+1} + \gamma V(S_{t+1}) - V(S_t)] \to 0$, καθώς $N(S_t) \to \infty$.

Ακόμη [24], υποθέτοντας ότι το βήμα ικανοποιεί τις συνθήκες Robbins-Monro: $\sum \alpha_t \to \infty$, $\sum \alpha_t^2 < \infty$, η ακολουθία των εκτιμήσεων $V_k \in \mathbb{R}^{|\mathcal{S}|}$, αποδεικνύεται ότι συγκλίνει ασυμπτωτικά στην πραγματική $v_\pi(s)$

Σε γενικές γραμμές, οι μέθοδοι MC χρησιμοποιούν μια εκτίμηση της (3.7) ως στόχο θεωρώντας ένα realisation της Gt μέσω δειγματοληψίας, ενώ οι $TD(0)$ μια εκτίμηση της (3.10) θεωρώντας realisation της R_{t+1} μέσω δειγματοληψίας και κάνοντας bootstrap από τη υπάρχουσα εκτίμηση $V(S_{t+1})$.

Algorithm 5: Πρόβλεψη TD(0)

Input: policy π, **Output**: V_π
Initialize $V(s)$ $\forall s$ (say 0)
for *each episode* **do**
  Initialize S
  for *each step of the episode* **do**
    A $\leftarrow$ action given by π for S
    Take action A, observe R and next state S'
    $V(S) \leftarrow V(S) + \alpha(R + \gamma V(S') - V(S))$
    S $\leftarrow$ S'

Παρατηρούμε ότι η αξία μιας κατάστασης ενημερώνεται βάση ενός δείγματος μετάβασης σε επόμενη κατάσταση, όπως στο Monte Carlo, και όχι βάση όλων των δυνατών επόμενων καταστάσεων, όπως το δυναμικό προγραμματισμό. Επίσης, όπως το δυναμικό προγραμματισμό, γίνεται bootstrap της αξίας της κατάστασης από την αξία της επόμενης κατάστασης(στη περίπτωση του δυναμικού προγραμματισμού την μέση σταθμισμένη αξία όλων των δυνατών επόμενων καταστάσεων).

Ένα από τα βασικά πλεονεκτήματα του TD σε σχέση με το MC είναι ότι τα updates στις συναρτήσεις αξίας γίνονται online μετά από κάθε βήμα, το οποίο ενδείκνυται για μη επεισοδιακά περιβάλλοντα.

Μια βασική διαφορά μεταξύ των δύο προσεγγίσεων της συνάρτησης αξίας είναι ότι η προσέγγιση τουMC μέσω της επιστροφής G_t συχνά είναι θορυβώδης και περιέχει υψηλή διακύμανση, δεδομένου ότι αθροίζουμε realisation τυχαίων μεταβλητών. Από την άλλη, η προσέγγιση της συνάρτησης αξίας μέσω TD έχει πολύ χαμηλότερη διακύμανση, αλλά βασίζεται σε προσέγγιση της αξίας της επόμενης κατάστασης(bootstrap) η οποία δεν είναι αμερόληπτη. Αυτή η διαφορά υπογραμμίζει το trade-off μεταξύ μεροληψίας(bias) και διακύμανσης(variance). Ο MC κάνει αμερόληπτες προσεγγίσεις με υψηλή διακύμανση, ενώ οι προσεγγίσεις του TD περιέχουν χαμηλή διακύμανση αλλά μεροληπτούν προς την προσέγγιση της συνάρτησης αξίας προς άλλες καταστάσεις που ενδεχομένως να είναι λανθασμένες.

Τέλος, θεωρούμε σταθερό batch K επεισοδίων T_k εμπειριών, από το οποίο καλούνται οι MC και TD να υπολογίσουν την συνάρτηση αξίας. Επαναλαμβάνοντας τα update των δύο αλγορίθμων επί των δεδομένων προκύπτουν δύο διαφορετικές λύσεις. Ο MC συγκλίνει στην συνάρτηση αξίας που ελαχιστοποιεί το μέσο τετραγωνικό σφάλμα $\sum_k \sum t(g_t^k - V(s_t^k))^2$. Αντίθετα, η μέθοδος TD, συγκλίνει στην λύση μέγιστης πιθανοφάνειας ΜΔΑ που εξηγεί τα δεδομένα [21].

Επειδή οι TD μέθοδοι χρησιμοποιούν την Μαρκοβιανή ιδιότητα μέσω του bootstrap, τε-

ίνουν να έχουν καλύτερες επιδόσεις σε σχέση με το MC ως αναφορά την ταχύτητα σύγκλισης σε περιβάλλοντα που την ικανοποιούν.

Αξιολογηση με TD n-βημάτων

Το χάσμα μεταξύ TD(0) και MC μπορεί να γεφυρωθεί από μια οικογένεια μεθόδων TD n-βημάτων. Ο MC εκτιμά την συνάρτηση αξίας χρησιμοποιώντας όλες τις μεταβάσεις από μια κατάσταση μέχρι το τέλος του επεισοδίου, ενώ ο TD(0) κάνει την εκτίμησή του για μια κατάσταση με βάση την μετάβαση από την κατάσταση αυτή στην επόμενη. Ορίζουμε τους στόχους TD(TD targets) μετά από n βήματα:

$$G_t^{(n)} = R_{t+1} + \gamma R_{t+2} + \ldots + \gamma^{n-1} R_{t+n} + \gamma^n V(S_{t+n}) \tag{3.38}$$

Στη περίπτωση που $t + n \geq T$, δηλαδή για τις μεταβάσεις στο τέλος του επεισοδίου, οι επιστροφές δίνονται όπως στο MC. Επίσης για $n \to \infty$, έχουμε την επιστροφή σύμφωνα με το MC. Έτσι, η επιστροφή MC αποτελεί μια ειδική περίπτωση της (3.38).

Με τον στόχο της εξίσωσης (3.38), ένα update παίρνει τη μορφή:

$$V(S_t) \leftarrow V(S_t) + \alpha(G_t^{(n)} - V(S_t)) \tag{3.39}$$

Αξιολογηση με TD(λ)

Η υλοποίηση του TD n-βημάτων απαιτεί την εκτέλεση n-βημάτων προτού γίνει ένα update και συνεπώς δεν γίνεται online, γεγονός που περιορίζει την πρακτικότητά του.

Θεωρώντας σταθμισμένους μέσους όρους των επιστροφών n-βημάτων για διάφορα n, προκύπτει ένα μεγάλο εύρος αλγορίθμων. Ο αλγόριθμος TD(λ) σταθμίζει τις επιστροφές:

$$G_t^{\lambda} = (1 - \lambda) \sum_{n=1}^{\infty} \lambda^{n-1} G_t^{(n)} \tag{3.40}$$

Η εκτίμηση ενός βήματος που αντιστοιχεί στο TD(0) λαμβάνει το μεγαλύτερο βάρος $1-\lambda$, και τα βάρη για τις επιστροφές n βημάτων φθίνουν γεωμετρικά. Στην επιστροφή T βημάτων, που πρακτικά αντιστοιχεί στην επιστροφή MC δίνεται βάρος λ^{T+1}, έτσι ώστε τα βάρη να αθροίζουν σε 1. Για $\lambda = 0$, η εκτίμηση παίρνει τη μορφή του TD(0) ενός βήματος, ενώ για $\lambda = 1$, έχουμε την εκτίμηση MC. Τα update έχουν την μορφή της (3.39), όπου αντικαθιστούμε το στόχο με τα G_t^{λ}.

Όπως και στο MC, η υλοποίηση του TD(λ) με βάση την (3.40) απαιτεί την λήξη του επεισοδίου προτού γίνει ένα update και συνεπώς δεν γίνεται να υλοποιηθεί online ως έχει. Παρόλα αυτά, η online υλοποίηση του TD(λ) είναι δυνατή [22], αν θεωρήσουμε το ίχνος $E_t(s) \in \mathbb{R}$ για την επισκεψη στη κατάσταση s τη χρονική στιγμή t. Για κάθε βήμα, το ίχνος όλων των καταστάσεων που δεν επισκεφθήκαμε φθίνει κατά $\gamma\lambda$:

$$E_t(s) = \begin{cases} \gamma\lambda E_{t-1}(s) + 1, \text{για επίσκεψη } s = S_t \\ \gamma\lambda E_{t-1}(s), \text{για όλες τις υπόλοιπες καταστάσεις} \end{cases} \tag{3.41}$$

όπου γ ο παράγοντας έκπτωσης και λ ο παράγοντας στάθμισης των επιστροφών n-βημάτων. Για την κατάσταση S_t που επισκεφθήκαμε τη χρονική στιγμή t, το ίχνος επιπλέον αυξάνεται κατά 1.

Διαισθητικά, το ίχνος μιας κατάστασης s διατηρεί ευρετικές για το πόσο πρόσφατα και πόσο συχνά συνέβησαν επισκέψεις στην κατάσταση s και ανάλογα ανανεώνεται η συνάρτηση αξίας της s.

3.4.3 Έλεγχος Monte Carlo

Όπως είδαμε, για την εφαρμογή της γενικευμένης βελτίωσης πολιτικής απαιτείται η γνώση της συνάρτησης αξίας κατάστασης-δράσης. Για τον υπολογισμό της συνάρτησης αξίας κατάστασης-δράσης, ακολουθούμε παρόμοια διαδικασία. Συγκεκριμένα για τον first-visit Monte Carlo, θεωρούμε τον μέσο όρο επιστροφών από την πρώτη φορά εντός του επεισοδίου που βρεθήκαμε στην κατάσταση s και επιλέξαμε την δράση a. Αντίστοιχα, για τον every-visit, θεωρούμε τις επιστροφές από κάθε φορά εντός του επεισοδίου που βρεθήκαμε στην κατάσταση s και επιλέξαμε την δράση a. Οι μέθοδοι αυτοί συγκλίνουν στην πραγματική $q_\pi(s,a)$ καθώς $N(s,a) \to \infty$.

Το πρόβλημα με αυτή τη προσέγγιση είναι ότι για ντετερμινιστική πολιτική π, σε κάθε κατάσταση s, επιλέγεται μόνο μια δράση και έτσι είναι αδύνατη η προσέγγιση της συνάρτησης αξίας με μέσους όρους. Ιδανικά, θα θέλαμε να γνωρίζουμε την συνάρτηση αξίας δράσης-κατάστασης για όλες τις δράσεις από μια κατάσταση s και όχι μόνο για την δράση που θεωρούμε βέλτιστη κάθε φορά.

Ένας τρόπος να σιγουρευτούμε ότι εξερευνούμε όλα τα ζεύγη καταστάσεων-δράσεων είναι να θεωρήσουμε επεισόδια που ξεκινούν από συγκεκριμένο ζεύγος κατάστασης-δράσης και κάθε ζεύγος κατάστασης-δράσης έχει μη μηδενική πιθανότητα να επιλεγεί ως αρχή του επεισοδίου. Αυτή η μέθοδος αναφέρεται ως exploring starts, και παρά το γεγονός ότι εξασφαλίζει συνεχή εξερεύνηση, έχει μικρή πρακτική χρησιμότητα σε πραγματικά περιβάλλοντα.

Ο πιο διαδεδομένος τρόπος να επιτύχουμε συνεχή εξερεύνηση είναι να θεωρήσουμε χαλαρές μη ντετερμινιστικές πολιτικές $\pi(a|s) > 0 \forall s \in \mathcal{S}, a \in \mathcal{A}$, και σταδιακά να κινούμαστε προς τη ντετερμινιστική βέλτιστη πολιτική. Η πολιτική που χρησιμοποιούμε ονομάζεται ε-άπληστη και έχει τη μορφή:

$$\pi(a|s) = \begin{cases} \frac{\epsilon}{m} + 1 - \epsilon, \text{ για } a = \underset{a'}{\operatorname{argmax}}[q(s,a')] \\ \frac{\epsilon}{m}, \text{αλλιώς} \end{cases} \tag{3.42}$$

όπου $\epsilon > 0$ μικρή παράμετρος της επιλογής μας. Σύμφωνα με αυτή τη πολιτική, επιλέγουμε μια τυχαία δράση με μικρή πιθανότητα ϵ και την βέλτιστη δράση με πιθανότητα $1 - \epsilon$.

Τέλος, για τον πρόβλημα του ελέγχου χρησιμοποιούμε την ιδέα της γενικευμένης βελτίωσης πολιτικής. Θεωρώντας πολιτική π με συνάρτηση $q_\pi(s,a)$, αποδεικνύεται [22] ότι η πολιτική π' που δρά ε-άπληστα ως προς την $q_\pi(s,a)$ αποτελεί βελτίωση ως προς την π. Αν η βελτίωση σταματήσει, τότε έχει επιτευχθεί η βέλτιστη πολιτική και η βέλτιστη συνάρτηση κατάστασης-δράσης.

Στα παραπάνω δεν δόθηκαν λεπτομέρειες για αξιολόγηση πολιτικής δηλαδή τον υπολογισμό της συνάρτησης αξίας της αρχικής πολιτικής, το οποίο αποτελεί το πρώτο μέρος της γενικευμένης βελτίωσης πολιτικής. Οι μέθοδοι αξιολόγησης πολιτικής του προηγούμενου μέρους μπορούν να χρησιμοποιηθούν προς αυτό το σκοπό.

Παρά το γεγονός ότι αποδείξαμε ότι η ε-άπληστη βελτίωση πολιτικής συγκλίνει στη βέλτιστη ε-άπληστη πολιτική, πρακτικά ενδιαφερόμαστε για την άπληστη ντετερμινιστική πολιτική π^* που ικανοποιεί την εξίσωση βελτιστότητας Bellman. Αποδεικνύεται[21], ότι αν όλα τα ζεύγη καταστάσεων-δράσεων επιλέγονται απείρως συχνά $\lim k \to \infty N(s,a) = \infty$, $\forall(s,a)$ και η πολιτική συγκλίνει σε άπληστη $\epsilon \to 0$, τότε η μέθοδος βελτίωσης πολιτικής με χρήση MC για την αξιολόγηση πολιτικής συγκλίνει στην βέλτιστη πολιτική π^* και βέλτιστη συνάρτηση αξίας q^*. Μια επιλογή για το ϵ είναι $\epsilon = 1/k$, όπου k το τρέχον επεισόδιο. Η μέθοδος αυτή ονομάζεται Άπληστη στο όριο με Άπειρη εξερεύνηση MC(GLIE-MC)

Algorithm 6: Έλεγχος Monte Carlo

Output: π^*, Q^*

Initialize Q for every state-action pair randomly

repeat

  Sample an episode $\{S_0, A_0, R_1, ..., S_T\}$ using π

  for *each pair* (S_t, A_t) *in the episode* **do**

    $N(S_t, A_t) \leftarrow N(S_t, A_t) + 1$

    $Q(S_t, A_t) \leftarrow Q(S_t, A_t) + \frac{1}{N(S_t, A_t)}(G_t - Q(S_t, A_t))$

  Improve policy by acting ϵ-greedy w.r.t. new Q

until *Forever*;

3.4.4 Έλεγχος Sarsa και Sarsa(λ)

Αν αντικαταστήσουμε την αξιολόγηση πολιτικής MC στη γενικευμένη βελτίωση πολιτικής με τις μεθόδους TD που εξετάσαμε προηγούμενα παίρνουμε μια καινούργια κατηγορία αλγορίθμων ελέγχου που ονομάζονται SARSA και SARSA(λ). Στο SARSA(λ) χρησιμοποιούμε τα ίχνη για τον TD(λ) για online ενημέρωση.

Όπως και στο MC αποδεικνύεται ότι [21], αν ικανοποιούνται οι συνθήκες GLIE και επιπλέον οι συνθήκες Robins-Monro $\sum \alpha_t \to \infty$, $\sum \alpha_t^2 < \infty$ για το βήμα α τότε οι SARSA συγκλίνουν στην συνάρτηση αξίας.

3.4.5 Off-policy έλεγχος-Q-learning

Σε όλες τις παραπάνω μεθόδους θεωρήσαμε ότι η πολιτική που ακολουθείται από τον πράκτορα είναι ίδια με την πολιτική που αξιολογούμε. Τα αντίστοιχα προβλήματα πρόβλεψης και ελέγχου ονομάζονται on-policy. Αντίθετα, οι περιπτώσεις όπου έχουμε ένα σύνολο επεισοδίων που παρήχθησαν με μια πολιτική $\mu(a|s)$ και ενδιαφερόμαστε για την αξιολόγηση ή βελτίωση μιας διαφορετικής πολιτικής $\pi(a|s)$ ονομάζονται off policy.

Algorithm 7: Έλεγχος Sarsa

Output: π^*, Q^*

Initialize Q for every state-action pair randomly

repeat

  Initialize S

  Choose A from S using policy derived from Q (e.g., ϵ-greedy)

  for *each step in the episode* **do**

    Take action A, observe R, S'

    Choose A' from S' using policy derived from Q (e.g., ϵ-greedy)

    $Q(S, A) \leftarrow Q(S, A) + \alpha[R + \gamma Q(S', A') - Q(S, A)]$

    $S \leftarrow S'$, $A \leftarrow A'$

until *for every episode*;

Algorithm 8: Έλεγχος Sarsa(λ)

Output: π^*, Q^*

Initialize Q for every state-action pair randomly

repeat

  Initialize traces $E(s, a) \leftarrow 0 \forall s \in \mathcal{S}, a \in \mathcal{A}$

  Initialize S

  Choose A from S using policy derived from Q (e.g., ϵ-greedy)

  for *each step in the episode* **do**

    Take action A, observe R, S'

    Choose A' from S' using policy derived from Q (e.g., ϵ-greedy)

    $\delta \leftarrow R + \gamma Q(S', A') - Q(S, A)$

    $E(S, A) \leftarrow E(S, A) + 1$

    for *every state* **do**

      $Q(s, a) \leftarrow Q(s, a) + \alpha\delta E(s, a)$

      $E(s, a) \leftarrow \gamma\lambda E(s, a)$

    $S \leftarrow S'$, $A \leftarrow A'$

until *for every episode*;

Για να χρησιμοποιηθούν επεισόδια που παρήχθησαν υπό την πολιτική μ, πρέπει να εξασφαλίζεται ότι όλες οι δράσεις που μπορούν να επιλεγούν υπό την π, μπορούν επίσης να επιλεγούν από την μ. Η πιο συνηθισμένη και ενδιαφέρουσα περίπτωση είναι η πολιτική μ από την οποία παράγονται τα επεισόδια να είναι στοχαστική και να επιδίδεται σε εξερεύνηση του χώρου καταστάσεων - δράσεων(πολιτική συμπεριφοράς) για παράδειγμα ε-άπληστη, ενώ η πολιτική π να είναι η άπληστη ντετερμινιστική πολιτική ως προς την τρέχουσα εκτίμηση της συνάρτησης αξίας(πολιτική στόχος). Αυτή η περίπτωση αποτελεί ένα από τους πιο γνωστούς αλγορίθμους στην ενισχυτική μάθηση, το Q-learning. Η ανανέωση για ένα βήμα παίρνει τη μορφή:

$$Q(S,A) \leftarrow Q(S,A) + \alpha(R + \gamma \max_{A'} Q(S',A') - Q(S,A)) \tag{3.43}$$

Όπου οι δράσεις A_t λαμβάνονται σύμφωνα με ε-άπληστη πολιτική σε όλα τα βήματα, αλλά το bootstrap γίνεται θεωρώντας ότι θα επιλεγεί δράση A' από την άπληστη πολιτική $A' = argmax(a')Q(S',a')$. Σημειώνεται ότι, η δράση από την οποία γίνεται bootstrap δεν είναι απαραίτητα η δράση που θα επιλεγεί στο επόμενο βήμα. Για αυτό το λόγο ο αλγόριθμος Q-learning θεωρείται off-policy αλγόριθμος.

Algorithm 9: Έλεγχος Q-learning

Output: π^*, Q^*

Initialize Q for every state-action pair randomly

repeat

  Initialize S

  Choose A from S using policy derived from Q (e.g., ϵ-greedy)

  for *each step in the episode* **do**

    Take action A, observe R, S'

    Choose A' from S' using policy derived from Q (e.g., ϵ-greedy)

    $Q(S,A) \leftarrow Q(S,A) + \alpha(R + \gamma \max_{A'} Q(S',A') - Q(S,A))$

    $S \leftarrow S'$, $A \leftarrow A'$

until *for every episode*;

Κεφάλαιο 4

Ενισχυτική Μάθηση με προσέγγιση

Στο κεφάλαιο 3, εξετάσαμε αλγορίθμους ενισχυτικής μάθησης όπου μπορούμε να αποθηκεύουμε τις τιμές της συνάρτησεις αξίας σε πίνακες. Σε πραγματικά προβλήματα, με μεγάλο πλήθος καταστάσεων και δράσεων ανά κατάσταση, κάτι τέτοιο δεν είναι δυνατό, όχι μόνο για λόγους μνήμης αλλά και λόγω του γεγονότος ότι οι περισσότερες καταστάσεις δεν θα επισκεφθούν επαρκή αριθμό φορών ώστε να να κάνουμε ακριβείς προβλέψεις περί της αξίας τους.

Προς αυτό το σκοπό, μπορούμε να χρησιμοποιήσουμε νευρωνικά δίκτυα, και την ικανότητα τους να γενικεύουν σε παραδείγματα που δεν έχουν ξαναδεί, ώστε να πάρουμε καλές προσεγγίσεις για καταστάσεις που βλέπουμε για πρώτη φορά. Συγκεκριμένα, δεδομένου ότι τα νευρωνικά δίκτυα είναι παγκόσμιοι προσεγγιστές, μπορούμε να τα χρησιμοποιήσουμε για να μάθουμε συναρτήσεις αξίας κατάστασης ή κατάστασης-δράσης, ή ακόμα και την ίδια την πολιτική, χρησιμοποιώντας εμπειρίες του πράκτορα. Το είδος της συνάρτησης που προσεγγίζει το νευρωνικό δίκτυο καθορίζει τις δύο οικογένειες μεθόδων της βαθιάς ενισχυτικής μάθησης: βασισμένη στη συνάρτηση αξίας και βασισμένη στη πολιτική.

4.1 Ενισχυτική Μάθηση βασισμένη στη συνάρτηση αξίας

Στην ενισχυτική μάθηση βασισμένη στη συνάρτηση αξίας, χρησιμοποιούνται προσεγγιστές όπως τα νευρωνικά δίκτυα για την εκτίμηση της συνάρτησης αξίας κατάστασης-δράσης και έπειτα ε-άπληστες πολιτικές για την επιλογή δράσεων. Η πολιτική δεν μοντελοποιείται explicitly.

4.1.1 Αξιολόγηση πολιτικής

Για το πρόβλημα της πρόβλεψης της συνάρτησης αξίας $\hat{v}$ χρησιμοποιούμε ένα νευρωνικό δίκτυο με σύνολο παραμέτρων θ. Δεδομένου ότι θέλουμε να προβλέψουμε μια συνεχή συνάρ-

τηση αξίας, χρησιμοποιούμε συνάρτηση κόστους του νευρωνικού δικτύου που αντιστοιχεί σε προβλήματα παλινδρόμησης (2.4).

$$\mathcal{J}(\theta) = \mathbb{E}_{s\sim\pi}[(v_\pi^{target}(s) - \hat{v}(s;\theta))^2] \tag{4.1}$$

όπου v_π^{target} είναι η επιθυμητή τιμή στόχος της συνάρτησης αξίας σύμφωνα με την πολιτική π που χρησιμοποιείται για την δειγματοληψία δράσεων. Για την εκμάθηση της συνάρτησης $\hat{v}$ χρησιμοποιούμε τον αλγόριθμο μέγιστης κατάβασης (ή τους αλγορίθμους βελτιστοποίησης που εξετάσαμε στο κεφάλαιο 2)

$$\Delta\theta = -\frac{1}{2}\alpha\frac{\partial\mathcal{J}}{\partial\theta} \tag{4.2}$$

όπου α ο ρυθμός μάθησης του νευρωνικού δικτύου.

Όπως είδαμε στο κεφάλαιο των νευρωνικών δικτύων, μπορούμε να προσεγγίσουμε την συνάρτηση $\mathcal{J}$ χρησιμοποιώντας μόνο ένα παράδειγμα(SGD):

$$\tilde{\mathcal{J}}(\theta) = (v_\pi^{target}(S_t) - \hat{v}(S_t;\theta))^2 \tag{4.3}$$

Και ο αλγόριθμος μέγιστης κλίσης παίρνει τη μορφή:

$$\Delta\theta = \alpha(v_\pi^{target}(S_t) - \hat{v}(S_t;\theta))\frac{\partial\hat{v}(S_t;\theta)}{\partial\theta} \tag{4.4}$$

Γενικά ο στόχος v_π^{target} που χρησιμοποιούμε είναι άγνωστος. Μπορούμε όμως να τον εκτιμήσουμε από εμπειρίες χρησιμοποιώντας τις εκτιμήσεις MC, TD και TD(λ) του κεφαλαίου 3. Έτσι προκύπτουν τα παρακάτω updates:

$$\Delta\theta = \alpha(G_t - \hat{v}(S_t;\theta))\frac{\partial\hat{v}(S_t;\theta)}{\partial\theta} \tag{4.5}$$

$$\Delta\theta = \alpha(R_{t+1} + \gamma\hat{v}(S_{t+1};\theta) - \hat{v}(S_t;\theta))\frac{\partial\hat{v}(S_t;\theta)}{\partial\theta} \tag{4.6}$$

$$\Delta\theta = \alpha(G_t^\lambda - \hat{v}(S_t;\theta))\frac{\partial\hat{v}(S_t;\theta)}{\partial\theta} \tag{4.7}$$

Αξίζει να σημειωθεί ότι η σύγκλιση των αλγορίθμων στην συνάρτηση αξίας είναι εγγυημένη για γραμμικούς προσεγγιστές και TD(0). Αντίθετα στην περίπτωση του TD με μη-γραμμικούς προσεγγιστές, όπως τα νευρωνικά δίκτυα, η σύγκλιση δεν εγγυάται.

4.1.2 Έλεγχος On-Policy

Για το πρόβλημα του ελέγχου χρησιμοποιούμε την ιδέα της γενικευμένης βελτίωσης πολιτικής του κεφαλαίου 3. Το νευρωνικό δίκτυο χρησιμοποιείται για την προσέγγιση της συνάρτησης αξίας κατάστασης-δράσης $\hat{q}(s,a;\theta)$ με στόχους MC, TD, TD(λ) αντίστοιχα:

$$\Delta\theta = \alpha(G_t - \hat{q}(S_t,A_t;\theta))\frac{\partial\hat{q}(S_t,A_t;\theta)}{\partial\theta} \tag{4.8}$$

$$\Delta\theta = \alpha(R_{t+1} + \gamma\hat{q}(S_{t+1}, A_t; \theta) - \hat{q}(S_t, A_t; \theta))\frac{\partial\hat{q}(S_t, A_t; \theta)}{\partial\theta} \tag{4.9}$$

$$\Delta\theta = \alpha(G_t^{\lambda} - \hat{q}(S_t, A_t; \theta))\frac{\partial\hat{q}(S_t, A_t; \theta)}{\partial\theta} \tag{4.10}$$

και στην συνέχεια χρησιμοποιούμε ε-άπληστη πολιτική για την βελτίωση της πολιτικής.

Στα παραπάνω χρησιμοποιήσαμε SGD για την προσέγγιση της συνάρτησης κόστους με ένα μόνο παράδειγμα. Αυτή η μέθοδος εκτός από τα μειονεκτήματα του SGD που παρουσιάστηκαν στο κεφάλαιο 2, έχει το επιπλέον πρόβλημα ότι χρησιμοποιεί το κάθε παράδειγμα μόνο μια φορά.

Αντί αυτού, μπορούμε να χρησιμοποιήσουμε την ιδέα των mini-batches και να ανανεώνουμε τις παραμέτρους του νευρωνικού δικτύου σε εποχές, χρησιμοποιώντας έτσι κάθε παράδειγμα τόσες φορές όσο και το πλήθος των εποχών. Η μέθοδος αυτή κρίνεται αναγκαία σε πραγματικά προβλήματα, καθώς η βαθιά ενισχυτική μάθηση χρειάζεται μεγάλο αριθμό δεδομένων για να επιτύχει υψηλές αποδόσεις.

Συγκεκριμένα, χρησιμοποιούμε την υπάρχουσα πολιτική π για τη δειγματοληψία εμπειριών (S_t, A_t, R_{t+1}) για T βήματα και κατασκευάζουμε ένα σύνολο δεδομένων $D = \{(S_t, A_t, Q_{\pi}^{target}(S_t, A_t))\}$, όπου τα $Q_{\pi}^{target}(S_t, A_t)$ υπολογίζονται όπως τις εξισώσεις (4.8), (4.9), (4.10), ορίζοντας τη συνάρτηση κόστους:

$$\mathcal{J}(\theta) = \mathbb{E}_{(s,a)\sim D}[(Q_{\pi}^{target}(s, a) - \hat{q}(s, a; \theta))^2] \tag{4.11}$$

4.1.3 Έλεγχος Off-Policy με Deep Q-Network

Ίσως ο πιο διαδεδομένος αλγόριθμος βαθιάς ενισχυτικής μάθησης είναι το DQN [12]. Πρόκειται για αλγόριθμο off-policy ελέγχου, παρόμοιο με το Q-learning της ενότητας 3.4.5 που χρησιμοποιεί βαθιά νευρωνικά δίκτυα για την προσέγγιση της συνάρτησης αξίας κατάστασης-δράσης. Συγκεκριμένα, για εμπειρία $(S_t, A_t, R_{t+1}, S_{t+1})$, ο στόχος προς τον οποίο ανανεώνονται οι παράμετροι του δικτύου είναι:

$$Q^{target}(S_t, A_t) = R_{t+1} + \gamma \max_{a'} \hat{q}(S_{t+1}, a'; \theta) \tag{4.12}$$

Η αφελής χρήση των παραπάνω δομικών στοιχείων έχει παρατηρηθεί ότι δημιουργεί αστάθεια στην εκπαίδευση βαθιών νευρωνικών δικτύων. Ένας λόγος για την ασταθή συμπεριφορά της εκπαίδευσης είναι η χρονική συσχέτιση μεταξύ διαδοχικών δεδομένων, δηλαδή το γεγονός ότι τα δεδομένα μας δεν είναι ανεξάρτητα και όμοια κατανεμημένα (i.i.d.). Ένας ακόμη λόγος είναι η συσχέτιση μεταξύ στόχου και πρόβλεψης. Από την εξίσωση (4.12) παρατηρούμε ότι κάθε φορά που ανανεώνουμε τις παραμέτρους του δικτύου ως προς το στόχο, αλλάζουμε και τον ίδιο το στόχο. Αυτό αναφέρεται στην βιβλιογραφία ως πρόβλημα κινούμενου στόχου και μπορεί να οδηγήσει τα updates των παραμέτρων του δικτύου σε απόκλιση.

Για την επίλυση των παραπάνω προβλημάτων, στο [12] χρησιμοποιούνται δύο βασικές μέθοδοι. Η μία αφορά την αποσυσχέτιση μεταξύ των δεδομένων μέσω ενός μηχανισμού μνήμης που

ονομάζεται Μνήμη Επανάληψης Εμπειριών(Experience Replay Memory). Σύμφωνα με αυτό, διατηρούν στην μνήμη ένα μεγάλο πλήθος εμπειριών, στο οποίο οι καινούργιες εμπειρίες αντικαθιστούν τις παλαιότερες, και για κάθε ανανέωση των παραμέτρων δειγματοληπτούν τυχαία εμπειρίες από την μνήμη, αφαιρώντας τη πιθανή συσχέτιση μεταξύ δεδομένων. Ταυτόχρονα, χρησιμοποιούνται batch updates για την εξομάλυνση της συνάρτησης κόστους του δικτύου και, δεδομένου ότι τα παραδείγματα που δειγματοληπτούνται προέρχονται από παλαιότερες πολιτικές, επιτυγχάνεται εξομάλυνση αναφορικά με τις πολιτικές.

Για την αντιμετώπιση του προβλήματος του κινούμενου στόχου, χρησιμοποιείται ένα δεύτερο νευρωνικό δίκτυο, που ονομάζεται target Q-network, το οποίο αποτελεί μια παλαιότερη έκδοση του δικτύου που εκπαιδεύουμε και διατηρείται σταθερό για μεγάλα χρονικά διαστήματα. Έτσι επιτυγχάνεται αποσύζευξη μεταξύ εκπαιδευόμενου δικτύου και στόχου.

Τέλος, όπως και στην αντίστοιχη περίπτωση του κεφαλαίου 3, οι δράσεις σε κάθε βήμα επιλέγονται με βάση ε-άπληστη πολιτική επί του εκπαιδευόμενου δικτύου $\hat{q}$.

Συνοπτικά, για μνήμη εμπειριών $D = \{(s, a, r, s')\}$, η συνάρτηση κόστους έχει τη μορφή:

$$\mathcal{J}(\theta) = \mathbb{E}_{(s,a,r,s')\sim D}[(r + \gamma \max_{a'} q(s', a'; \theta_{old}) - \hat{q}(s, a; \theta))^2] \tag{4.13}$$

όπου θ_{old} οι παράμετροι του δικτύου στόχου.

4.1.4 Επεκτάσεις του Deep Q-Network

Έχουν προταθεί πολλές παραλλαγές και επεκτάσεις στο κλασικό DQN με τις σημαντικότερες να αναφέρονται παρακάτω.

double Q-learning

Σύμφωνα με την εξίσωση (4.13), ο τελεστής max χρησιμοποιείται τόσο για την επιλογή της επόμενης κατάστασης όσο και για την εκτίμηση του bootstrap. Αυτό μπορεί να επαληθευτεί εύκολα αν γράψουμε την εξίσωση ως:

$$Q^{target}(S_t, A_t) = R_{t+1} + \gamma q(S_{t+1}, \operatorname*{argmax}_{a'} q(S_{t+1}, a'; \theta_{old}); \theta_{old}) \tag{4.14}$$

Αυτό μπορεί να προκαλέσει υπερεκτίμηση της συνάρτησης αξίας, έχοντας ως αποτέλεσμα τον λανθάνοντα οπτιμισμό. Αυτό το πρόβλημα επεκτείνεται όταν κάνουμε bootstrap από μια τιμή η οποία είναι ήδη υπερβολικά οπτιμιστική.

Για την αντιμετώπιση του προβλήματος μπορούμε να αποσυζεύξουμε τις διαδικασίες επιλογής και εκτίμησης στην εξίσωση, χρησιμοποιώντας 2 συναρτήσεις αξίας κατάστασης δράσης θ, θ_{old} [6]. Αυτή η ιδέα χρησιμοποιείται και στις tabular περιπτώσεις του κεφαλαίου 3, χρησιμοποιώντας δύο πίνακες q_1, q_2, ανανεώνοντας κάθε φορά τυχαία τον ένα πίνακα κάνοντας bootstrap από τις τιμές του άλλου.

Στα πλαίσια του doubleDQN [6], χρησιμοποιείται το εκπαιδευόμενο δίκτυο για την επιλογή της επόμενης δράσης και το δίκτυο στόχο για την εκτίμηση της συνάρτησης αξίας-κατάστασης, όπως στην εξίσωση:

$$Q^{target}(S_t, A_t) = R_{t+1} + \gamma q(S_{t+1}, \underset{a'}{\text{argmax}}\hat{q}(S_{t+1}, a'; \theta); \theta_{old}) \tag{4.15}$$

Όπως και στο Q-learning του κεφαλαίου 3, θεωρούμε ότι στο επόμενο βήμα θα ακολουθήσουμε την άπληστη πολιτική, όμως η πραγματική εκτίμηση γίνεται με βάση τη συνάρτηση στόχο.

Μνήμη επανάληψης εμπειρίας με προτεραιότητα

Μια ακόμη επέκταση αφορά τη δειγματοληψία της μνήμης με προτεραιότητα [18]. Η βασική ιδέα είναι ότι θα θέλαμε να δειγματοληπτούμε πιο συχνά τα παραδείγματα από τα οποία μπορούμε να μάθουμε περισσότερο ή αλλιώς τα παραδείγματα στα οποία παρατηρούνται τα περισσότερα λάθη. Αυτή η ιδέα χρησιμοποιείται συχνά και στο πλαίσιο της επιβλεπόμενης μάθηση, όπου στα παραδείγματα με την μεγαλύτερη συνάρτηση λάθους δίνεται μεγαλύτερο βάρος στάθμισης στο τελικό υπολογισμό του κόστους.

Ως κριτήριο για την δειγματοληψία ενός παραδείγματος από την μνήμη εμπειριών χρησιμοποιείται το σφάλμα TD της εξίσωσης (3.33), το οποίο στο πλαίσιο των doubleDQN (4.15) μπορεί να εκφραστεί για μια εμπειρία j ως:

$$\delta_j = Q^{target}(S_j, A_j) - Q(S_j, A_j) \tag{4.16}$$

Η κατανομή με την οποία δειγματοληπτούνται τα παραδείγματα, θα πρέπει να είναι τέτοια ώστε όλα τα παραδείγματα να έχουν μη μηδενική πιθανότητα επιλογής και ταυτόχρονα η πιθανότητα επιλογής τους να αυξάνεται με το σφάλμα TD. Αυτό σημαίνει ότι δεν μπορεί να υπάρξει αυστηρή ταξινόμηση της μνήμης εμπειριών, καθώς κάποια παραδείγματα ενδέχεται να μην επιλεγούν ποτέ κατά τη διάρκεια ύπαρξής τους στην μνήμη.

Μια επιλογή που πληροί τις παραπάνω προϋποθέσεις είναι η πιθανότητα επιλογής μιας εμπειρίας να είναι ανάλογη με το td-error. Πρακτικά, επειδή ανανεώνουμε συνεχώς τις παραμέτρους του δικτύου, το td-error και άρα η προτεραιότητα όλων των εμπειριών που βρίσκονται στη μνήμη, θα πρέπει να ανανεώνεται. Αυτό είναι κοστοβόρο και αντί αυτού, επιλέγεται να ανανεώνεται το td-error και άρα η προτεραιότητα μόνο για τις εμπειρίες που επιλέχθηκαν για το batch εκπαίδευσης. Μια συνέπεια αυτής της λεπτομέριας υλοποίησης, σε συνδυασμό με την ανάλογη με το td-error πιθανότητα δειγματοληψίας είναι ότι, οι εμπειρίες με αρχικά χαμηλό td-error, ενδέχεται να μην επιλεγούν ποτέ κατά τη διάρκεια ύπαρξής τους στην μνήμη. Ακόμη, δεδομένου ότι κατά την εκπαίδευση των νευρωνικών δικτύων το κόστος φθίνει αργά, οι εμπειρίες με αρχικά υψηλό td-error επιλέγονται πολύ πιο συχνά, γεγονός που μπορεί να οδηγήσει σε υπερπροσαρμογή του δικτύου σε αυτές. Έτσι επιλέγεται ως πιθανότητα δειγματοληψίας μιας εμπειρίας i η:

$$P(i) = \frac{\delta_i^\alpha}{\sum_k \delta_k^\alpha} \tag{4.17}$$

όπου $0 \leq \alpha \leq 1$ υπερπαράμετρος της επιλογής μας. Αυτή η κατανομή πιθανότητας επιτυγχάνει μια ισορροπία μεταξύ της αναλογικής δειγματοληψίας($\alpha = 1$) και της ομοιόμορφης τυχαίας δειγματοληψίας($\alpha = 0$), πληρώντας ταυτόχρονα τις προαναφερθείσες προϋποθέσεις

περί μονοτονίας με την προτεραιότητα και μη μηδενική πιθανότητα. Το γεγονός, ότι η κατανομή με την οποία επιλέγουμε παραδείγματα εμπειριών για ένα batch είναι διαφορετική από την κατανομή που χρησιμοποιείται για την παραγωγή των παραδειγμάτων, δηλαδή την πολιτική, σημαίνει ότι εισάγεται μεροληψία. Για την διόρθωση μεροληψίας, χρησιμοποιείται η τεχνική του importance sampling:

$$\mathbb{E}_{X\sim P}[f(X)] = \mathbb{E}_{X\sim Q}[\frac{P(X)}{Q(X)}f(X)] \tag{4.18}$$

Σταθμίζοντας τα td-error που χρησιμοποιούνται στην οπισθοδιάδοση του λάθους με βάρη:

$$w_i = (\frac{1}{N}\frac{1}{P(i)})^{\beta} \tag{4.19}$$

όπου N το μέγεθος της μνήμης και η παράμετρος $0 \leq \beta \leq 1$ καθορίζει το μέγεθος αυτής της διόρθωσης($\beta = 0$ καθόλου διόρθωση, $\beta = 1$ πλήρη διόρθωση). Επίσης, τα βάρη κανονικοποιούνται στο διάστημα $(0, 1)$ για λόγους ευστάθειας.

Αναφορικά με την υλοποίηση, η δειγματοληψία ανάλογη με μια προτεραιότητα(ή, όπως εξηγήθηκε παραπάνω, ανάλογη με τη προτεραιότητα υψωμένη σε μια δύναμη α) μπορεί να υλοποιηθεί με μια επιλογή ρουλετας. Θεωρούμε N δείγματα, με προτεραιότητες p_i^{α} και θέλουμε να δειγματοληπτήσουμε με πιθανότητα $P(i)$ (4.17). Σχηματίζουμε τα συσσωρευτικά αθροίσματα $s_i = \sum_{k<i} P(k)$, και παράγουμε ένα τυχαίο αριθμό r στο διάστημα $[0, 1]$. Με δυαδική αναζήτηση στη λίστα s, μπορούμε να βρούμε το στοιχείο των N δειγμάτων $\hat{i}$ για το οποίο $\hat{s}_{i-1} < r < \hat{s}_i$. Αυτή η διαδικασία παρομοιάζεται με το γύρισμα ρουλέτας με slots μεγέθους ανάλογου με την προτεραιότητα. Η δειγματοληψία έχει πολυπλοκότητα $O(logn)$ ενώ η κατασκευή της λίστας s, $O(n)$, η οποία είναι απαγορευτική για την περίπτωση της μνήμης εμπειριών με προτεραιότητα στην οποία καινούργια δείγματα καταφθάνουν συνεχών αντικαθιστώντας τα παλαιότερα.

Αντί αυτού χρησιμοποιείται μια δομή παρόμοια με τη σωρό, με τη διαφορά ότι ο κόμβος του πατέρα έχει ως τιμή το άθροισμα των παιδιών. Ως φύλλα του δέντρου τοποθετούμε τις προτεραιότητες των εμπειριών της μνήμης. Αυτή η δομή ονομάζεται δέντρο άθροισης διαστημάτων(sum-segment tree) και όπως στη περίπτωση της σωρού μπορεί να υλοποιηθεί ως λίστα και έχει τη μορφή.

$$S_{node} = \begin{cases} p_i, \text{ για κόμβους φύλλα} \\ S_{left\_child} + S_{right\_child}, \text{ για εσωτερικούς κόμβους} \end{cases} \tag{4.20}$$

Η εισαγωγή μιας καινούργιας εμπειρίας στη μνήμη περιλαμβάνει την άμεση αντικατάσταση του παλαιότερου φύλλου και επανυπολογισμό των αθροισμάτων των κόμβων προγόνων του με πολυπλοκότητα ανάλογη με το ύψος του δέντρου, $O(logn)$. Για την δειγματοληψία, επιλέγεται τυχαίος αριθμός p_r στο διάστημα $[0, ptotal]$, όπου $ptotal$ η τιμή του κορμού του δέντρου, και κατά βάθος διάσχιση του με τρόπο ανάλογο της δυαδικής αναζήτησης.

Όπως και στη περίπτωση της ρουλέτας αναζητούμε $\hat{i}$ τέτοιο ώστε

$$\sum_{i=0}^{\hat{i}} a_i \leq p_r \leq \sum_{i=0}^{\hat{i}+1} a_i \tag{4.21}$$

Ξεκινώντας από το κορμό του δέντρου ελέγχουμε την (4.21) μέσω των παιδιών του. Συγκεκριμένα, αν $s_{root}(left) = \sum_{i=0}^{\frac{N}{2}} a_i > p_r$, τότε αναγκαστικά $\hat{i} < \frac{N}{2}$ και κινούμαστε προς το αριστερό υποδέντρο. Αλλιώς, $\hat{i} \geq \frac{N}{2}$ και κινούμαστε προς το δεξί. Στο επόμενο βήμα πρέπει να μπορούμε να ελέγξουμε την (4.21). Αν κινηθήκαμε αριστερά, κάτι τέτοιο είναι εύκολο καθώς έχουμε ήδη στην διάθεσή μας το άθροισμα $\sum_{i=0}^{\frac{N}{4}} a_i$ στο αριστερό παιδί του τρέχοντος κόμβου. Αν είχαμε κινηθεί δεξιά τότε το αριστερό παιδί του τρέχοντος κόμβου θα έχει το άθροισμα $\sum_{i=\frac{N}{2}}^{\frac{3N}{4}} a_i$, ενώ χρειαζόμαστε το $\sum_{i=0}^{\frac{3N}{4}} a_i = \sum_{i=0}^{\frac{N}{2}} a_i + \sum_{i=\frac{N}{2}}^{\frac{3N}{4}} a_i$. Η λογική αυτή επεκτείνεται για οποιοδήποτε εσωτερικό κόμβο του δέντρου. Συνεπώς, κάθε φορά που κινούμαστε προς το δεξί υποδέντρο, αφαιρούμε τη τιμή του αριστερού παιδιού του τρέχοντος κόμβου. Η διαδικασία τερματίζεται όταν φτάσουμε σε κόμβο φύλλο.

Algorithm 10: double Qlearning with prioritized experience replay

Input: minibatch k, learning rate η, replay period K and size N, exponents α and β, total time steps T.

Initialize replay memory D empty, set max priority to 1
Initialize action-value function Q with random weights θ
Initialize target action-value function Q-target with weights θ_{old}
Observe S_0 and choose A_0 from S_0 using policy derived from Q (e.g., ϵ-greedy)
for $t = 0, \ldots, T$ **do**
    Observe S_{t+1}, R_{t+1}
    Store transition $(S_t, A_t, R_{t+1}, S_{t+1})$ in D with max priority
    if $t \mod K = 0$ **then**
        for $j = 1, \ldots, k$ **do**
            Sample transition j using (4.17)
            Compute importance sample weights w_j, (4.19)
            Compute Q_j^{target} using (4.15)
            Append tuple $(S_j, A_j, Q_j^{target}, w_j)$ in batch B
        Adam Update network parameters with L_2-norm cost using batch B
        for $j = 1, \ldots, k$ **do**
            Compute new td-error δ_j from (4.16)
            Update priorities $p_j \leftarrow |\delta_j|$ and update sample's parents in sum-segment tree
        Empty batch B
        Every once in a while update the target network parameters $\theta_{old} \leftarrow \theta$
    choose A_t from S_t using policy derived from Q (e.g., ϵ-greedy)

Αρχιτεκτονική μονομαχίας

Συχνά, για την καλύτερη προσέγγιση της συνάρτησης αξίας κατάστασης-δράσης Q, πρέπει να είμαστε σε θέση να διαφοροποιήσουμε μεταξύ της αξίας επιλογής μιας δράσης από μια συγκεκριμένη κατάσταση από την αξία της κατάστασης αυτής. Ορίζουμε ως πλεονέκτημα της

δυάδας (s, a) την συνάρτηση $A(s, a)$, για την οποία ισχύει:

$$Q(s, a) = V(s) + A(s, a) \tag{4.22}$$

Το πλεονέκτημα $A(s, a)$ εκφράζει την αξία που επιτυγχάνεται μέσω της επιλογής της δρασης a από την κατάσταση s, αγνοώντας το πόσο καλή είναι η κατάσταση s, δηλαδή το $V(s)$. Η συνάρτηση πλεονεκτήματος πετυχαίνει την αποσύζευξη μεταξύ της αξίας της κατάστασης-δράσης $Q(s, a)$ και την αξία της κατάστασης $V(s)$ και εξασφαλίζει ότι ο πράκτορας επιλέγει την βέλτιστη δράση.

Η προτεινόμενη αρχιτεκτονική ονομάζεται ανταγωνιστική [26] και αποτελείται από δύο ξεχωριστά κανάλια υπολογισμών για την συνάρτηση αξίας κατάστασης και τα πλεονεκτήματα για κάθε δράση. Τα κανάλια συνδυάζονται στην έξοδο για να παράξουν της συνάρτηση αξίας κατάστασης-δράσης αφήνοντας ίδιες τις υπόλοιπες λεπτομέρειες του DQN. Το βασικό πλεονέκτημα της αρχιτεκτονικής αυτής είναι ότι μπορεί να μάθει ποιές καταστάσεις έχουν υψηλή αξία χωρίς να χρειάζεται να μάθει την επίδραση όλων των δράσεων από αυτή τη κατάσταση, ειδικά όταν η επιλογή οποιασδήποτε δράσης είναι ασήμαντη.

Για την παραγωγή της συνάρτησης $Q(s, a)$ από τα κανάλια υπολογισμού της συνάρτησης αξίας και των πλεονεκτημάτων για κάθε δράση, λαμβάνουμε υπόψη ότι:

$$V_\pi(s) = \mathbb{E}_{a\sim\pi}[Q_\pi(s, a)] \tag{4.23}$$

Και άρα

$$\mathbb{E}_{a\sim\pi}[A_\pi(s, a)] = 0 \tag{4.24}$$

Επίσης, για άπληστη πολιτική που επιλέγει $a^* = \underset{\alpha}{\text{argmax}} Q(s, a)$ πρέπει:

$$Q(s, a^*) = V(s) \Longrightarrow A(s, a^*) = 0 \tag{4.25}$$

Στα πλαίσια της εκπαίδευσης ενός δικτύου με τη συνάρτηση στόχο Q^{target} να δίνεται για την παραγωγή του κόστους και την εφαρμογή του αλγορίθμου οπισθοδιάδοσης, πρέπει να εγγυάται ότι, για την δράση που επιλέχθηκε πρέπει να ισχύει η (4.25). Επιπλέον, δίνοντας την συνάρτηση Q^{target}, δεν εγγυόμαστε ότι τα δύο κανάλια υπολογίζουν την συνάρτηση αξίας κατάστασης και τα πλεονεκτήματα για κάθε δράση. Θα μπορούσαν για παράδειγμα να υπολογίζουν εκδοχές αυτών μετατοπισμένες κατά μια σταθερά προς τα πάνω και προς τα κάτω αντίστοιχα. Εν ολίγοις, στη εξίσωση (4.22), δεδομένης της Q δεν μπορούμε να βρούμε μοναδικά V, A.

Για την αντιμετώπιση αυτού του προβλήματος, αντί για την άμεση χρήση της (4.22) για την παραγωγή της συνάρτησης αξίας κατάσταση-δράσης, χρησιμοποιείται η:

$$Q(s, a) = V(s) + A(s, a) - \max_{a'} A(s, a') \tag{4.26}$$

Έτσι, για την άπληστη δράση $a^* = \underset{\alpha}{\text{argmax}} Q(s, a)$, εγγυόμαστε ότι τα κανάλια προσεγγίζουν ακριβώς τις συναρτήσεις αξίας κατάστασης και το αντίστοιχο πλεονέκτημα.

Μια εναλλακτική που χρησιμοποιείται στην πράξη είναι η:

$$Q(s,a) = V(s) + A(s,a) - \frac{1}{|\mathcal{A}|}\sum_{a'} A(s,a') \tag{4.27}$$

η οποία επιτυγχάνει παρόμοια αποτελέσματα και παρατηρήθηκε ότι βελτιώνει την ευστάθεια στην διαδικασία εκπαίδευσης [26], καθώς με την (4.27) τα πλεονεκτήματα χρειάζεται να ενημερώνονται τόσο γρήγορα όσο το μέσο όρο, αντί να αντισταθμίζουν μεγάλες αλλαγές στη συνάρτηση πλεονεκτήματος για τις άπληστες δράσεις στην (4.26).

4.2 Ενισχυτική μάθηση βασισμένη στη πολιτική

Στα προηγούμενα, εξετάσαμε μεθόδους ενισχυτικής μάθησης που βασίζονται στην προσέγγιση της συνάρτησης αξίας ή αξίας κατάστασης-δράσης. Με βάση την προσέγγιση αυτή η επιλογή δράσεων γίνεται μέσω άπληστων ή ε-άπληστων πολιτικών.

Οι μέθοδοι αυτοί έχουν το βασικό μειονέκτημα ότι συγκλίνουν σε ντετερμινιστικές πολιτικές, ενώ σε πολλά περιβάλλοντα, η βέλτιστη πολιτική είναι στοχαστική, όπως για παράδειγμα το παιχνίδι πέτρα-ψαλίδι-χαρτί. Ένα ακόμα μειονέκτημα είναι ότι μικρές αλλαγές στην συνάρτηση αξίας ενός ζεύγους κατάστασης-δράσης κατά την διάρκεια της εκπαίδευσης, μπορεί να το κάνουν να μην επιλεγεί ποτέ στο μέλλον, γεγονός που εισάγει ασυνέχειες που οδηγούν στην αδυναμία σύγκλισης για προσεγγιστές όπως τα νευρωνικά δίκτυα. Για παράδειγμα, αλγόριθμοι όπως το Q-learning, Sarsa έχουν παρατηρηθεί να μην συγκλίνουν σε κάποια πολιτική, όταν χρησιμοποιούνται μη-γραμμικοί προσεγγιστές [23]

Οι βασική ιδέα των προσεγγιστικών μεθόδων που βασίζονται στην πολιτική είναι η μοντελοποίηση και προσέγγιση μιας στοχαστικής πολιτικής με νευρωνικά δίκτυα. Η τυπική μοντελοποίηση περιλαμβάνει βαθιά νευρωνικά δίκτυα, όπου τα αρχικά επίπεδα μεταφέρουν τις εμπειρίες σε ένα χώρο χαρακτηριστικών, ενώ το τελικό επίπεδο λαμβάνει ως είσοδο την αναπαράσταση της κατάστασης στο χώρο αυτό και παράγει μια κατανομή πιθανότητας που εκφράζει την πιθανότητα επιλογής μιας δράσης, μέσω χρήσης της $softmax$ συναρτησης ενεργοποίησης του κεφαλαίου 2.

Η επιλογή της συνάρτησης κόστους για επεισοδιακά περιβάλλοντα με παράγοντα έκπτωσης $0 \leq \gamma \leq 1$ εκφράζει τη μέση επιστροφή:

$$\mathcal{J}(\theta) = \mathbb{E}_{\pi_\theta}[\sum_t \gamma^t R_{t+1}] \tag{4.28}$$

όπου θ οι παράμετροι της πολιτικής $\pi(s,a;\theta) = P[A_t = a|S_t = s;\theta]$. Οι συναρτήσεις αξίας δίνονται κατά τα γνωστά από τις (3.7), (3.8).

Η (4.28) γράφεται:

$$\begin{aligned}\mathcal{J}(\theta) =& \mathbb{E}_{\pi_\theta}[\sum_t \gamma^t R_{t+1}] = \sum_t \gamma^t \mathbb{E}_{\pi_\theta}[R_{t+1}] \\ =& \sum_s (\sum_{s_0}\sum_t \gamma^t p_0(s_0) p(S_0 \to s, t, \pi_\theta)) \sum_a \pi(s,a;\theta) R_s^a\end{aligned} \tag{4.29}$$

Όπου $p_0(S_0)$ η πιθανότητα το επεισόδιο να ξεκινά από την κατάσταση $s0$ και $p(S_0 \rightarrow s, t, \pi_\theta)$ η πιθανότητα το περιβάλλον να μεταβαίνει από την κατάσταση s_0 στην κατάσταση s_t μετά από t βήματα ακολουθώντας την πολιτική π. Ορίζοντας

$$d^{\pi_\theta}(s) = \sum_{s_0} \sum_{t} \gamma^t p_0(s_0) p(S_0 \rightarrow s, t, \pi_\theta) \tag{4.30}$$

ως το εκπτωτούμενο στάθμισμα των πιθανοτήτων επισκέψεων στις καταστάσεις s_t, η συνάρτηση κόστους γράφεται

$$\mathcal{J}(\theta) = \sum_{s} d^{\pi_\theta}(s) \sum_{a} \pi(s, a; \theta) R_s^a \tag{4.31}$$

Στη περίπτωση των μη-επεισοδιακών καταστάσεων, η συνάρτηση κόστους έχει επίσης τη μορφή (4.31), όπου το $d^{\pi_\theta}(s)$ αντιπροσωπεύει τη στάσιμη κατανομή πιθανότητας εύρεσης στην κατάσταση s.

Για την μεγιστοποίηση των συναρτήσεων $\mathcal{J}$ ως προς τις παραμέτρους θ, χρησιμοποιούμε τον αλγόριθμο μέγιστης κλίσης ανάβασης:

$$\Delta\theta = \alpha \frac{\partial \mathcal{J}}{\partial \theta} \tag{4.32}$$

Αποδεικνύεται [23] ότι και για τις δύο επιλογές συνάρτησης κόστους ισχύει:

$$\frac{\partial \mathcal{J}}{\partial \theta} = \sum_{s} d^{\pi_\theta}(s) \sum_{a} \nabla_\theta \pi(s, a; \theta) Q(s, a) \tag{4.33}$$

Αυτό το αποτέλεσμα αποτελεί το θεώρημα κλίσης πολιτικής(Policy gradient. Παρατηρούμε ότι δεν εμφανίζεται ο όρος $\nabla_\theta d^{\pi_\theta}(s)$, δηλαδή η επίδραση των αλλαγών της πολιτικής μέσω παραμέτρων θ στην κατανομή επίσκεψης των καταστάσεων. Αυτό μας επιτρέπει την προσέγγιση των κλίσεων με δειγματοληψία. Γράφοντας

$$\nabla_\theta \pi(s, a; \theta) = \pi(s, a; \theta) \nabla_\theta \log \pi(s, a; \theta) \tag{4.34}$$

Η (4.33) γράφεται:

$$\begin{aligned} \frac{\partial \mathcal{J}}{\partial \theta} =& \sum_{s} d^{\pi_\theta}(s) \sum_{a} \pi(s, a; \theta) \nabla_\theta \log \pi(s, a; \theta) Q_\pi(s, a) \\ =& \mathbb{E}_{\pi_\theta}[\nabla_\theta \log \pi(s, a; \theta) Q_\pi(s, a)] \end{aligned} \tag{4.35}$$

Η (4.35) επιτρέπει την άμεση εφαρμογή του αλγορίθμου μέγιστης κλίσης ανάβασης, υπολογίζοντας τον όρο εντός της προσδοκώμενης τιμής από παραδείγματα. Παρόλα αυτά ο όρος $Q_\pi(s, a)$ πρέπει να εκτιμηθεί καθώς δεν είναι γνωστός.

4.2.1 Ο αλγόριθμος REINFORCE

Μια προσέγγιση του $Q_\pi(s, a)$ είναι μέσω της επιστροφής G_t ακολουθώντας την προσέγγιση Monte Carlo. Αυτή αποτελεί τη βάση του αλγορίθμου REINFORCE, ο οποίος αποτελείται από δειγματοληψία της πολιτικής για το μήκος ενός επεισοδίου και SGD updates.

Algorithm 11: REINFORCE

Initialize policy network with random weights θ

for *every episode* **do**

  Draw an episode $\{S_0, A_0, R_1, S_1, \ldots, S_{T-1}, A_{T-1}, R_T, S_T\} \sim \pi_\theta$

  for *every step t of episode* **do**

    Calculate G_t

    Update θ with gradients: $\frac{\partial \mathcal{J}}{\partial \theta} = \nabla_\theta \log \pi(S_t, A_t; \theta) G_t$

Η προσέγγιση της συνάρτησης αξίας κατάστασης-δράσης από δείγματα της επιστροφής είναι αμερόληπτη. Το πρόβλημα αυτής της μεθόδου, είναι το ίδιο με το γενικότερο πρόβλημα των μεθόδων MC και αφορά τη πολύ υψηλή διακύμανση των εκτιμίσεων της Q μέσω των επιστροφών.

4.2.2 Ο αλγόριθμος Δράστη-Κριτή

Η μέθοδος αυτή αφορά την εκτίμηση της συνάρτησης $Q_\pi(s, a)$ μέσω ενός προσεγγιστή με παραμέτρους w: $Q_\pi(s, a) \approx Q(s, a; w)$. Ο προσεγγιστής με παραμέτρους w ονομαζεται κριτής και αναλαμβάνει την προσέγγιση της συνάρτησης αξίας κατάστασης-δράσης από εμπειρίες. Ο προσεγγιστής δράστης αναλαμβάνει το *update* της πολιτικής σύμφωνα με τις προσεγγίσεις του κριτή και εμπειρίες ακολουθώντας την (4.35).

Για προσέγγιση της $Q_\pi(s, a)$ μέσω $Q(s, a; w)$ θεωρούμε το κόστος:

$$\mathcal{E}(w) = \mathbb{E}_{\pi_\theta}[(Q_\pi(s, a) - Q(s, a; w))^2] \tag{4.36}$$

Για την εκπαίδευση του κριτή, οι παράμετροι w επιλέγονται για την ελαχιστοποίηση του κόστους, ακολουθώντας την κλίση μέγιστης κατάβασης. Όταν ο αλγόριθμος συγκλίνει, θα πρέπει:

$$\begin{aligned} \nabla_w \mathcal{E}(w) &= \nabla_w \mathbb{E}_{\pi_\theta}[(Q_\pi(s, a) - Q(s, a; w))^2] = 0 \\ \mathbb{E}_{\pi_\theta}[(Q_\pi(s, a) - Q(s, a; w)) \nabla_w Q(s, a; w)] &= 0 \end{aligned} \tag{4.37}$$

Αν ικανοποιείται η (4.37) και επιπλέον η προσέγγιση $Q(s, a; w)$ είναι συμβατή με τη πολιτική:

$$\nabla_w Q(s, a; w) = \nabla_\theta \log \pi_\theta(s, a; \theta) \tag{4.38}$$

τότε η (4.37) γράφεται:

$$\mathbb{E}_{\pi_\theta}[(Q_{\pi_\theta}(s, a) \nabla_\theta \log \pi_\theta(s, a; \theta)] = \mathbb{E}_{\pi_\theta}[(Q(s, a; w) \nabla_\theta \log \pi_\theta(s, a; \theta)] \tag{4.39}$$

και άρα η εκπαίδευση της πολιτικής (4.35) μπορεί να πάρει τη μορφή:

$$\frac{\partial \mathcal{J}}{\partial \theta} = \mathbb{E}_{\pi_\theta}[\nabla_\theta \log \pi(s, a; \theta) Q(s, a; w)] \tag{4.40}$$

Το αποτέλεσμα αυτό αποτελεί τον θεώρημα Policy Gradient with Function Approximation [23]. Για την εκπαίδευση του κριτή από εμπειρίες μπορούν να χρησιμοποιηθούν μέθοδοι

όπως MC και TD για την παραγωγή των στόχων στη συνάρτηση κόστους. Η σύγκλιση των ανανεώσεων δράστη-κριτή σε τοπικό ελάχιστο της συνάρτησης κόστους αποδεικνύεται στο [23].

4.2.3 Δράστης-Κριτής Πλεονεκτήματος και ο αλγόριθμος A2C

Συχνά, επιλέγεται η αφαίρεση μιας συνάρτησης βάσης, η οποία εξαρτάται μόνο από την κατάσταση s, $B(s)$ από την (4.40) όπως:

$$\frac{\partial \mathcal{J}}{\partial \theta} = \mathbb{E}_{\pi_\theta}[\nabla_\theta \log \pi(s,a;\theta)(Q_\pi(s,a;w) - B(s))] \tag{4.41}$$

Η χρήση της $B(s)$ βοηθά στη μείωση της διακύμανσης των εκτιμήσεων των κλίσεων [23] καθώς $Var(X-Y) = Var(X)+Var(Y)-2Cov(X,Y)$, χωρίς να εισάγει επιπλεον μεροληψία αφού:

$$\mathbb{E}_{\pi_\theta}[\nabla_\theta \log \pi(s,a;\theta)B(s)] = \sum_s \delta^{\pi_\theta}(s)B(s)\nabla_\theta \sum_a \pi_\theta(s,a;\theta) = 0 \tag{4.42}$$

Η πιο διαδεδομένη επιλογή είναι η χρήση της συνάρτησης αξίας κατάστασης $V_\pi(s)$. Έτσι η (4.41), χρησιμοποιώντας την (4.22) γράφεται:

$$\frac{\partial \mathcal{J}}{\partial \theta} = \mathbb{E}_{\pi_\theta}[\nabla_\theta \log \pi(s,a;\theta)A(s,a;w)] \tag{4.43}$$

Όπου $A(s,a)$ η συνάρτηση πλεονεκτήματος. Στη περίπτωση αυτή, μιλάμε για το αλγόριθμο δράστη-κριτή πλεονεκτήματος, όπου ο δράστης μαθαίνει τη πολιτική και ο κριτής τη βάση $V_\pi(s)$. Όπως και πριν μπορούμε να χρησιμοποιήσουμε μεθόδους MC και TD για την παραγωγή του στόχου του κριτή. Μια προσέγγιση, ανάλογη με τη TD(λ) του κεφαλαίου 3, είναι η γενικευμένη εκτίμηση πλεονεκτήματος(GAE) [19]. Δεδομένων ενός batch από εμπειρίες $(s_t, a_t, r_{t+1}, s_{t+1})$, ορίζουμε το πλεονέκτημα n βημάτων για την χρονική στιγμή t:

$$A_t^{(n)} = \sum_{l=0}^{n-1} \gamma^l R_{t+l+1} + \gamma^n V(S_{t+n}) - V(S_t) \tag{4.44}$$

Το σταθμισμένο πλεονέκτημα A_t^λ, χρησιμοποιώντας τα td-errors ενός βήματος $\delta_t = R_{t+1} + \gamma V(S_{t+1}) - V(S_t)$ μπορεί να γραφεί:

$$\begin{aligned} A_t^\lambda =& (1-\lambda)(A_t^{(1)} + \lambda A_t^{(2)} + \ldots) \\ =& (1-\lambda)(\delta_t + \lambda(\delta_t + \gamma\delta_{t+1})) + \lambda^2(\delta_t + \gamma\delta_{t+1} + \gamma^2\delta_{t+2}))\ldots) \\ =& (1-\lambda)(1+\lambda+\lambda^2+\ldots)(\delta_t + \gamma\lambda\delta_{t+1} + (\gamma\lambda)^2\delta_{t+2} + \ldots) \\ =& \sum_{l=0}^{\infty} (\gamma\lambda)^l \delta_{t+l} \end{aligned} \tag{4.45}$$

Αντίστοιχα, για την σταθμισμένη επιστροφή G_t^λ:

$$\begin{aligned} G_t^\lambda =&(1-\lambda)(G_t^{(1)} + \lambda G_t^{(2)} + \ldots) \ , \ G_t^{(n)} = A_t^{(n)} + V(S_t) \\ =&(1-\lambda)(A_t^{(1)} + V(S_t) + \lambda(\ A_t^{(2)} + V(S_t)) + \lambda^2(A_t^{(3)} + V(S_t)) + \ldots) \\ =&(1-\lambda)(A_t^{(1)} + \lambda A_t^{(2)} + \ldots) + (1-\lambda)(1+\lambda+\lambda^2+\ldots)V(S_t) \\ =&A_t^\lambda + V(S_t) \end{aligned} \tag{4.46}$$

Algorithm 12: A2C

Input: rollout length n, number of environments N_e, total time steps T, γ, λ
Initialize actor π and critic V with parameters θ, θ_v respectively
repeat
  for *actor* $1, \ldots, N_e$ **do**
    for *time* $t, \ldots, t+n$ **do**
      Sample actions a from policy π_θ
      Execute actions a and observe rewards r and next states s'
      Store transition (s, a, r, s')
    Compute A_t^λ, G_t^λ from (4.45), (4.46)
  Make batch D by concatenating $(s_j, a_j, A_j^\lambda, G_j^\lambda)$ from every actor
  Adam Update network parameters θ, θ_v on batch D by minimizing cost (4.50)
until T;

Στη πράξη [11], [14] χρησιμοποιούμε βαθιά νευρωνικά δίκτυα για την μοντελοποίηση της πολιτικής π_θ και της συνάρτησης V_{θ_v}. Τα δίκτυα μοιράζονται παραμέτρους στα πρώιμα επίπεδα της αρχιτεκτονικής και έπειτα χωρίζονται σε δύο κανάλια. Το στάδιο εξόδου του καναλιού της πολιτικής έχει συνάρτηση ενεργοποίησης softmax για την παραγωγή μιας κατανομής πιθανότητας. Το στάδιο εξόδου του δικτύου της συνάρτησης αξίας αποτελείται από ένα νευρώνα και έχει συνάρτηση με γραμμική συνάρτηση ενεργοποίησης.

Μια σημαντική διαφορά από τις μεθόδους συνάρτησης αξίας είναι ότι δεν χρησιμοποιείται μνήμη επανάληψης εμπειρίας. Η ανεξαρτησία μεταξύ παραδειγμάτων επιτυγχάνεται με χρήση πολλαπλών περιβαλλόντων. Το θετικό αυτής της μεθόδου είναι ότι με πολλά περιβάλλοντα είναι δυνατό να εξερευνόνται διαφορετικές περιοχές του χώρου καταστάσεων. Έτσι, επιτυγχάνεται μεγάλη ποικιλία στα παραδείγματα που παράγονται από τα διάφορα περιβάλλοντα για ένα batch, γεγονός που σταθεροποιεί την διαδικασία εκπαίδευσης με τρόπο ανάλογο με αυτό της μνήμης εμπειριών. Επίσης, το γεγονός ότι δεν χρησιμοποιείται μνήμη εμπειριών για την εξασφάλιση ανεξαρτησίας μεταξύ παραδειγμάτων του ίδιου batch, επιτρέπει τη χρήση on-policy αλγορίθμων.

Η διαδικασία εκπαίδευσης περιλαμβάνει τη συγκέντρωση διαδοχικών εμπειριών $(s_t, a_t, r_{t+1}, s_{t+1})$ μήκους T από όλα τα περιβάλλοντα με χρήση της πολιτικής π_θ. Για κάθε περιβάλλον ξεχωριστά υπολογίζουμε τα GAE πλεονεκτήματα A_t^λ σύμφωνα με την (4.45), και τους στόχους του κριτή με την (4.46). Τέλος, δημιουργούμε το σύνολο εκπαίδευσης $(s_j, a_j, A_j^\lambda, G_j^\lambda)$ που αποτελείται από συγχώνευση των δεδομένων από όλα τα περιβάλλοντα σε ένα μεγάλο batch. Η συνάρτηση κόστους για τον κριτή και τον δράστη αντίστοιχα δίνεται από:

$$\mathcal{J}_{critic}(\theta_v) = \sum_j (G_j^\lambda - V(j;\theta_v))^2 \tag{4.47}$$

$$\mathcal{J}_{actor}(\theta) = \sum_j (A_j^\lambda \log \pi(j,a;\theta)) \tag{4.48}$$

με σκοπό την ελαχιστοποίηση για τον κριτή και τη μεγιστοποίηση για το δράστη. Για να εξασφαλίσουμε ότι ο αλγόριθμος δεν συγκλίνει πρόωρα σε μια πολιτική χρησιμοποιούμε ένα κόστος εντροπίας:

$$\mathcal{J}_{entropy}(\theta) = -\sum_j \sum_a \pi(j,a;\theta) \log \pi(j,a;\theta) \tag{4.49}$$

Η εντροπία αποτελεί ένα ΄μέτρο΄ της αβεβαιότητας που υπάρχει σχετικά με την πολιτική. Η μεγιστοποίηση του όρου αυτού εξασφαλίζει κατανομή που κατευθύνεται προς την ομοιόμορφη, γεγονός που συμβάλει στην εξερεύνηση. Η συνολική συνάρτηση κόστους προς ελαχιστοποίηση δίνεται:

$$\mathcal{J}_{total}(\theta,\theta_v) = c_1\mathcal{J}_{critic} - c_2\mathcal{J}_{actor} - c_3\mathcal{J}_{entropy} \tag{4.50}$$

Όπου c_1, c_2, c_3 υπερπαράμετροι της επιλογής μας. Τέλος, χρησιμοποιούμε τον αλγόριθμο Adam για τη βελτιστοποίηση των παραμέτρων.

4.2.4 Ο αλγόριθμος PPO

Ένα από τα προβλήματα των μεθόδων προσέγγισης της πολιτικής με νευρωνικά δίκτυα είναι ότι δεν υπάρχει τρόπος να ελέγξουμε το πως το μέγεθος των update στις παραμέτρους επηρεάζει την πολιτική. Στο [20] προτείνεται η φραγή της πολιτικής εντός μιας περιοχής εμπιστοσύνης που διασφαλίζει μικρές αλλαγές στη πολιτική μεταξύ διαδοχικών update. Αυτό επιτυγχάνεται με τη χρήση μιας εναλλακτικής συνάρτησης κόστους(surrogate objective function) για τα ζεύγη καταστάσεων-δράσεων (s,a) που αποτελείται από το λόγο της τρέχουσας πολιτικής για το ζεύγος (s,a) προς μια παλαιότερη έκδοση της πολιτικής για το (s,a).

Στο PPO [20], προτείνεται η χρήση της συνάρτησης κόστους δράστη:

$$\mathcal{J}_{actor}^{PPO} = \mathbb{E}[\min\{r_t(\theta)A_t,\ clip(r_t(\theta), 1-\epsilon, 1+\epsilon)A_t\} \tag{4.51}$$

όπου $\epsilon > 0$ μικρή σταθερά και:

$$r_t(\theta) = \frac{\pi(at|st,\theta)}{\pi_{old}(a_t|s_t,\theta_{old})} \tag{4.52}$$

Ο πρώτος όρος στην (4.51) είναι ο λόγος της καινούργιας προς την παλαιά πολιτική. Αρχικά ισούται με ένα και μετά το update αυξάνεται αν το πλεονέκτημα είναι θετικό, αλλιώς μειώνεται. Ο δεύτερος όρος περιορίζει το λόγο (4.52), κρατώντας την π κοντά στην π_{old}. Συγκεκριμένα, για θετικά πλεονεκτήματα κλιπάρουμε το λόγο πάνω ώστε να μην ενισχυθεί κατά πολύ η δράση της εμπειρίας, ενώ για αρνητικά πλεονεκτήματα κλιπάρουμε το λόγο προς

τα κάτω. Τέλος, παίρνουμε το ελάχιστο μεταξύ των δύο, ώστε η τελική συνάρτηση να είναι κάτω φράγμα του λόγου (4.52). Έτσι, αγνοούμε τις μεγάλες αλλαγές στο λόγο (4.52) αν φέρουν βελτίωση της συνάρτησης κόστους, ενώ τις συγκαταλέγουμε αν χειροτερεύουν την συνάρτηση κόστους.

Η διαδικασία εκπαίδευσης είναι παρόμοια με τον A2C με την διαφορά ότι χρησιμοποιούμε την (4.51) ως κόστος του δράστη. Επίσης, η βελτιστοποίηση της συνολικής συνάρτησης $\mathcal{J}_{total}$ γίνεται σε πολλές εποχές.

Algorithm 13: PPO

Input: rollout length n, number of environments N_e, total time steps T, γ, λ
Initialize actor π and critic V with parameters θ, θ_v respectively
repeat
  for *actor* $1, \ldots, N_e$ **do**
    for *time* $t, \ldots, t+n$ **do**
      Sample actions a from policy π_θ
      Execute actions a and observe rewards r and next states s'
      Store transition (s, a, r, s')
    Compute A_t^λ, G_t^λ from (4.45), (4.46)
  Make batch D by concatenating $(s_j, a_j, A_j^\lambda, G_j^\lambda)$ from every actor
  Adam Update network params w.r.t. J_{total}^{PPO} (4.50), (4.51) on batch D over K epochs with minibatch size m
until T;

4.3 Εξερεύνηση με ανταμοιβές περιέργειας

Μέχρι στιγμής, ο τρόπος με τον οποίο οι πράκτορες εξερευνούν καταστάσεις του περιβάλλοντος βασίζεται στην τύχη. Στη περίπτωση των τεχνικών βασισμένων στη συνάρτηση αξίας, χρησιμοποιούμε ε-άπληστες πολιτικές για να εξασφαλίσουμε την συνεχή εξερεύνηση καθόλη τη διάρκεια της εκπαίδευσης, ενώ στη περίπτωση των μεθόδων που βασίζονται στη πολιτική χρησιμοποιούμε ένα κόστος εντροπίας για να αποφύγουμε τη πρόωρη σύγκλιση σε ντετερμινιστικές πολιτικές.

Γενικά, σκοπός της ενισχυτικής μάθησης είναι η εκμάθηση πολιτικών που μεγιστοποιούν τις ανταμοιβές που δίνονται από το περιβάλλον. Σε πολλές πρακτικές εφαρμογές οι ανταμοιβές είναι σπάνιες ή η πυκνότητά τους μειώνεται όσο ο πράκτορας συνεχίζει να αλληλεπιδρά με το περιβάλλον, γεγονός που δημιουργεί πρόβλημα καθώς ο πράκτορας δεν είναι σε θέση να βελτιώσει τη πολιτική του παρα μόνο αν λάβει κάποιο σήμα ανταμοιβής από το περιβάλλον. Επίσης, σε πολλά περιβάλλοντα, η πιθανότητα ο πράκτορας να "σκοντάψει" πάνω σε μια ανταμοιβή αποκλειστικά με τυχαία εξερεύνηση είναι πολύ μικρή. Ακόμη, υπάρχουν περιβάλλοντα στα οποία οι ανταμοιβές του περιβάλλοντος είναι μεν πυκνές, αλλά η τυχαία εξερεύνηση δεν δύναται να βρει τις βέλτιστες πολιτικές εύκολα. Αυτά τα δύο κριτήρια μπορούν να ορίσουν

μια ταξινόμηση των περιβαλλόντων. Με κριτήριο το ποσό εξερεύνησης που απαιτείται για την εύρεση της βέλτιστης πολιτικής και με κριτήριο την πυκνότητα των ανταμοιβών.

Μια προσέγγιση στο πρόβλημα των σπάνιων ανταμοιβών είναι η παραγωγή ενός σήματος ανταμοιβής r^{int}, εμπνευσμένο από βιολογικούς οργανισμούς, το οποίο μοντελοποιεί τη περιέργεια ή το intrinsic motivation του πράκτορα για το περιβάλλον [16]. Οι μέθοδοι μοντελοποίησης που έχουν προταθεί κυμαίνονται σε δύο κατηγορίες. Στη πρώτη προσέγγιση, η μοντελοποίηση της περιέργειας ταυτίζεται με την ικανότητα ή μη του πράκτορα να προβλέψει την επόμενη κατάσταση του περιβάλλοντος δεδομένης της κατάστασης και της δράσης που έχει επιλέξει να εκτελέσει. Σε αυτή τη περίπτωση, απαιτείται η μοντελοποίηση της δυναμικής του περιβάλλοντος, και οι αντίστοιχες μέθοδοι ονομάζονται μέθοδοι δυναμικής. Η δεύτερη προσέγγιση, αφορά το προσδιορισμό της καινοτομίας(novelty) μιας κατάστασης σε σχέση με τις καταστάσεις που έχουν επισκεφθεί, το οποίο γενικά απαιτεί την γνώση μιας κατανομής πιθανότητας επί των διαφόρων καταστάσεων.

Ακόμη και σε περιβάλλοντα με πυκνό σήμα ανταμοιβών, η περιέργεια μπορεί να βοηθήσει στη εξερεύνηση του χώρου καταστάσεων με σκοπό την απόκτηση νέας γνώσης, η οποία είναι απαραίτητη για τη βελτίωση μιας εξερευνητικής πολιτικής και την εξισορρόπηση μεταξύ εξερεύνησης και εκμετάλλευσης [16]. Επίσης, ενδέχεται να οδηγήσει στη εκμάθηση δεξιοτήτων που μπορεί να είναι χρήσιμες σε μελλοντικά βήματα της αλληλεπίδρασης του πράκτορα με το περιβάλλον.

Σε κάθε περίπτωση, το σήμα περιέργειας σταθμίζεται με υπερπαράμετρο β και προστίθεται στο σήμα ανταμοιβής που παράγεται από το περιβάλλον για την παραγωγή της συνολικής ανταμοιβής: $r_{total} = r_{ext} + \beta r_{int}$. Το σήμα αυτό μπορεί να χρησιμοποιηθεί στην εκπαίδευση πρακτόρων με μεθόδους που περιγράφηκαν στις προηγούμενες ενότητες.

Ένα πρόβλημα που προκύπτει είναι ότι σε περιβάλλοντα στα οποία οι μεταβάσεις μεταξύ καταστάσεων υπόκεινται σε στοχαστικότητα, θα παράγεται υψηλό σήμα περιέργειας καθώς ο πράκτορας δεν θα είναι ποτέ σε θέση να προβλέψει την επόμενη κατάσταση του περιβάλλοντος στο πλαίσιο των μεθόδων που βασίζονται στη δυναμική. Αυτό ισχύει και στις μεθόδους που προσδιορίζουν τη καινοτομία μιας κατάστασης. Το πρόβλημα αναφέρεται με τον όρο noisy-TV. Ένας πράκτορας που παρατηρεί καταστάσεις οι οποίες είναι θορυβώδεις θα λαμβάνει συνεχώς σήμα περιέργειας και θα θέλει να τις επισκέπτεται συνεχώς. Άλλα παραδείγματα παρόμοιας στοχαστικότητας αφορούν την ύπαρξη στοιχείων του περιβάλλοντος που αποσπούν την προσοχή, δηλαδή επηρεάζουν το περιβάλλον με τρόπο που είναι δύσκολος να προβλεφθεί, αλλά η πρόβλεψη είναι πρακτικά άχρηστη για την λήψη αποφάσεων(π.χ. Το θρόισμα των φύλλων ενός δέντρου).

4.3.1 Ο αλγόριθμος ICM

Ο αλγόριθμος Intrinsic Exploration Module [16], ανήκει στη κατηγορία παραγωγής σήματος περιέργειας βασισμένη στη δυναμική. Συγκεκριμένα, δεδομένης μιας εμπειρίας $(s_t, a_t, r^{ext}_{t+1}, s_{t+1})$ χρησιμοποιούμε νευρωνικό δίκτυο φ που παράγει αναπαραστάσεις των καταστάσεων s_t, s_{t+1}. Οι αναπαράσταση της κατάστασης $\varphi(s_t)$ καθώς και η δράση που επιλέχθηκε a_t, τίθενται

ως είσοδοι σε ένα(η περισσότερα διαδοχικά) επίπεδο νευρώνων f που παράγει μια πρόβλεψη για την αναπαράσταση της επόμενης κατάστασης $\hat{\varphi}(s_{t+1})$. Σχηματίζοντας την L_2-νόρμα, μεταξύ της πρόβλεψης της αναπαράστασης της επόμενης κατάστασης και της πραγματικής αναπαράσταση της επόμενης κατάστασης παράγουμε το σήμα περιέργειας:

$$r_t^{int} = ||\hat{\varphi}(s_{t+1}) - \varphi(s_{t+1})||_2^2 \tag{4.53}$$

Το σήμα αυτό, μαζί με το r_t^{ext}, χρησιμοποιείται για την εκπαίδευση ενός δικτύου κριτή-δράστη με τον αλγόριθμο PPO.

Αναφορικά με την εκπαίδευση του μοντέλου αναπαράστασης, αυτό πρέπει να είναι σε θέση να ξεχωρίζει στοιχεία του περιβάλλοντος που είτε βρίσκονται υπό τον έλεγχο του πράκτορα είτε επηρεάζουν τη μετάβαση καταστάσεων του περιβάλλοντος και να αγνοεί στοιχεία όπως το θρόισμα των φύλλων ή το noisy-TV που εξετάσαμε παραπάνω.

Προς αυτό το σκοπό, χρησιμοποιείται ένα σχήμα αυτο-εκπαίδευσης στο οποίο εκπαιδεύουμε το νευρωνικό δίκτυο αναπαράστασης ώστε να προβλέπει την δράση $\hat{a}_t$, που επιλέχθηκε για την μετάβαση μεταξύ των καταστάσεων s_t, s_{t+1}. Δεδομένου ότι το δίκτυο πρέπει να προβλέψει τη δράση a_t, δεν έχει λόγο να αναπαριστά στοιχεία της κατάστασης του περιβάλλοντος που δεν επηρεάζουν τον πράκτορα. Η συνάρτηση κόστους μπορεί να διατυπωθεί ως το cross-entropy μεταξύ της προβλεπόμενης κατανομής $\hat{a}_t$ και της πραγματικής κατανομής στόχου με πιθανότητα 1 για την δράση a_t που επιλέχθηκε και 0 για τις υπόλοιπες δράσεις.

Πρακτικά, τα δύο δίκτυα συγχωνεύονται σε ένα ενιαίο δίκτυο με εισόδους τις καταστάσεις s_t, s_{t+1} και την δράση που έφερε τη μετάβαση μεταξύ αυτών a_t. Τα αρχικά επίπεδα, με παραμέτρους θ_ϕ, παράγουν τις αναπαραστάσεις $\phi(s_t;\theta_\phi)$, $\phi(s_{t+1};\theta_\phi)$. Το εμπρόσθιο μοντέλο f, με παραμέτρους θ_f και εισόδους $\phi(s_t;\theta_\phi)$, a_t, εκπαιδεύεται με συνάρτηση κόστους την

$$\mathcal{J}_{forward}(\theta_\phi, \theta_f) = \sum_t ||f(\varphi(s_t;\theta_\phi), a_t; \theta_f) - \varphi(s_{t+1})||_2^2 \tag{4.54}$$

Σημειώνεται ότι ο στόχος $\phi(s_{t+1})$ στο παραπάνω κόστος, παρά το γεγονός ότι εξαρτάται από την αναπαράσταση, θεωρείται σταθερά για την οπισθοδιάδοση των κλίσεων σφάλματος, δηλαδή $\nabla_{\theta_\phi}\phi(s_{t+1}) = 0$. Το οπίσθιο μοντέλο, με παραμέτρους θ_g, εκπαιδεύεται με συνάρτηση κόστους την

$$\mathcal{J}_{inverse}(\theta_\phi, \theta_g) = -\sum_t \sum_{a_t} a_t \log g(\phi(s_t;\theta_\phi), \phi(s_{t+1};\theta_\phi); \theta_g) \tag{4.55}$$

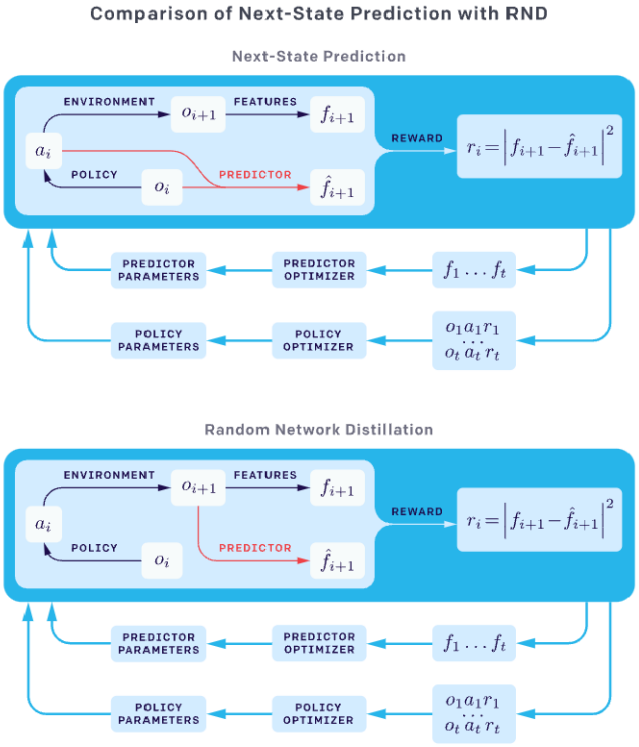
Η συνολική συνάρτηση κόστους δίνεται από:

$$\mathcal{J}_{icm}(\theta_\phi, \theta_f, \theta_g) = c\mathcal{J}_{forward}(\theta_\phi, \theta_f) + (1-c)\mathcal{J}_{inverse}(\theta_\phi, \theta_g) \tag{4.56}$$

όπου c υπερπαράμετρος της επιλογής μας. Η βελτιστοποίηση της συνάρτησης γίνεται από κοινού, ως προς όλες τις σχετικές παραμέτρους θ_ϕ, θ_f, θ_g.

4.3.2 Ο αλγόριθμος RND

Μια μέθοδος για την παραγωγή του σήματος περιέργιας που ανήκει στην κατηγορία μεθόδων ανίχνευσης καινοτομίας μιας κατάστασης είναι το Random Network Distillation [4].



Σχήμα 4.1: Διαφορές RND και μεθόδων πρόβλεψης επόμενης κατάστασης (Πηγή: [15])

Η μέθοδος περιλαμβάνει 2 νευρωνικά δίκτυα: ένα σταθερό και τυχαία αρχικοποιημένο δίκτυο στόχο f που παράγει μια τυχαία αναπαράσταση της κατάστασης μιας εμπειρίας και ένα δίκτυο πρόβλεψης $\hat{f}$ με παραμέτρους θ_f, το οποίο εκπαιδεύεται να προβλέπει τις εξόδους του στόχου, ελαχιστοποιώντας το κόστος ως προς θ_f:

$$\mathcal{J}_{rnd}(\theta_f) = \sum_t ||f(s_{t+1}) - \hat{f}(s_{t+1}, \theta_f)||_2^2 \tag{4.57}$$

Το παραπάνω κόστος χρησιμοποιείται ως ανταμοιβή εξερεύνησης r^{int}, καθώς αναμένουμε το κόστος να είναι υψηλό για καταστάσεις στις οποίες ο προβλέπτης δεν έχει εκπαιδευτεί. Έτσι ο αριθμός των φορών που έχει επισκεφθεί μια κατάσταση, μαθαίνεται implicitly: σε καταστάσεις που επισκέπτονται συχνά το λάθος πρόβλεψης θα είναι μικρό, ενώ σε καινοτόμες καταστάσεις μεγάλο. Δεδομένου ότι ο στόχος που παράγεται από το τυχαίο δίκτυο είναι ντετερμινιστικός, σημαίνει ότι μπορεί να μαθευτεί από το προβλέπτη μέσω κατάβασης μέγιστης κλίσης με το κόστος να προσεγγίζει ένα τοπικό ελάχιστο, ακόμη και για περιβάλλοντα με τα στοχαστικά στοιχεία που αναφέρθηκαν προηγούμενα.

Algorithm 14: Αλγοριθμος περιέργειας για ICM, RND

Input: rollout length n, number of environments N_e, total time steps T, γ, λ
Initialize actor π and critic V with parameters θ, θ_v respectively
Initialize curiosity network
repeat
  for *actor* $1, \ldots, N_e$ **do**
    for *time* $t, \ldots, t+n$ **do**
      Sample actions a from policy π_θ
      Execute actions a and observe rewards r^{ext} and next states s'
      Compute r^{int} from (4.53)
      $r \leftarrow \beta r^{int} + r^{ext}$
      Store transition (s, a, r, s')
    Compute A_t^λ, G_t^λ from (4.45), (4.46)
  Make batch D by concatenating $(s_j, a_j, A_j^\lambda, G_j^\lambda)$ from every actor
  Adam Update actor-critic network params w.r.t. J_{total}^{PPO} (4.50), (4.51) on batch D over K epochs with minibatch size m
  Adam Update curiosity network params on batch D over K epochs with minibatch size m using (4.56) or (4.57)
until T;

Κεφάλαιο 5

Περιγραφή και αξιολόγηση πειραμάτων

Η υλοποίηση των αλγορίθμων της παρούσας εργασίας έγινε σε Python. Η Python είναι μια γλώσσα υψηλού επιπέδου, η οποία χρησιμοποιείται ευρέως σε εφαρμογές μηχανικής μάθησης καθώς τα πιο διαδεδομένα πακέτα μοντελοποίησης νευρωνικών δικτύων, το Tensorflow και το Pytorch, παρέχουν εύχρηστα API σε Python.

Όλα τα νευρωνικά δίκτυα υλοποιήθηκαν με τη χρήση Tensorflow 2.x.. Το Tensorflow αποτελεί μια βιβλιοθήκη ανάπτυξης γραφικών μοντέλων, που βασίζεται στην αναπαράσταση πινάκων(tensors) και την ροή αυτών μέσα από τελεστές, δίνοντας την δυνατότητα για αυτόματη διαφόριση. Έτσι, για την εκπαίδευση ενός νευρωνικού δικτύου, αρκεί να οριστεί η συνάρτηση κόστους και οι παράμετροι του μοντέλου, χωρίς την explicit υλοποίηση του αλγορίθμου οπισθοδιάδοσης. Πιο συγκεκριμένα, το Tensorflow-keras παρέχει κλάσεις υψηλού επιπέδου για ευρέως χρησιμοποιούμενες πράξεις, όπως επίπεδα πλήρως συνδεδεμένων νευρώνων, συνελικτικά επίπεδα, συναρτήσεις λάθους, συναρτήσεις ενεργοποίησης και άλλα. Ο κώδικας μπορεί να βρεθεί στο https://github.com/marvlach/atari_rl

5.1 Gym Atari

Για την αξιολόγηση χρησιμοποιήθηκε η βιβλιοθήκη Gym[3]. Το Gym αποτελεί μια βιβλιοθήκη που παρέχει API για πληθος από διαφορετικά περιβάλλοντα, όπως επιτραπέζια παιχνίδια, κλασσικά βιντεοπαιχνίδια και περιβάλλοντα κλασσικού ελέγχου και ρομποτικής. Συγκεκριμένα χρησιμοποιήθηκε το Gym Atari, που παρέχει την εξομοίωση 49 διαφορετικών βιντεοπαιχνιδιών Atari.

Τα παιχνίδια Atari έχουν αποτελέσει σημαντικό benchmark για την ανάπτυξη και αξιολόγηση αλγορίθμων βαθιάς ενισχυτικής μάθησης. Πρόκειται για επεισοδιακά περιβάλλοντα. στα οποία παρέχεται η δυνατότητα θέσης της κατάστασης τους σε μια αρχική κατάσταση και αλλαγής της κατάστασής τους μέσω δράσεων. Οι δυνατές δράσεις αφορούν την χρήση ενός προσομοιωμένου joystick που ελέγχει τον χαρακτήρα κάθε παιχνιδιού καθώς και τη προσομοίωση ενός πλήκτρου fire. Ο εκτεταμένος χώρος δράσεων περιέχει 18 δράσεις((8 προ-

σανατολισμοί του joystick + μηδενικός προσανατολισμός) × πάτημα ή μη του πλήκτρου fire) και προσαρμόζεται εκ νέου για τις ανάγκες κάθε παιχνιδιού.

Η επικοινωνία του πράκτορα με το περιβάλλον γίνεται μέσω της κλάσης environment και των μεθόδων `reset`, `step`. Η μέθοδος `reset` επιστρέφει την κατάσταση του παιχνιδιού σε προκαθορισμένη αρχική κατάσταση. Η μέθοδος `step` εκτελεί ένα βήμα στο προσομοιωτή του παιχνιδιού. Δέχεται ως είσοδο μια εκ των δυνατών δράσεων και επιστρέφει την καινούργια παρατήρηση, το σήμα ανταμοιβής, και μια σημαία που αφορά το αν το παιχνίδι έλαβε τέλος(game over). Έτσι, οι πράκτορες μπορούν να δημιουργούν τετράδες εμπειριών $(s_t, a_t, r_{t+1}, s_{t+1})$ και να παίρνουν αποφάσεις.

Αναφορικά με τις καταστάσεις, το Gym επιστρέφει την εικόνα του παιχνιδιού σε RGB, το οποίο αποτελεί την τρέχουσα παρατήρηση. Εναλλακτικά, προσφέρει πρόσβαση στην εσωτερική κατάσταση της RAM του παιχνιδιού, χαρακτηριστικό το οποίο σπάνια χρησιμοποιείται.

Μια ταξινόμηση των παιχνιδιών που παρέχονται στο Gym φαίνεται στο πίνακα 5.1. Αρχικά, τα παιχνίδια ταξινομούνται με βάση το πόσο εύκολη είναι η επίτευξη υψηλού score μέσω πολιτικών τοπικής εξερεύνησης(όπως ε-άπληστες). Τα εύκολα παιχνίδια μπορούν να κατηγοριοποιηθούν εκ νέου βάσει του αν μπορούν να βρεθούν exploits που μεγιστοποιούν το score χωρίς να πετυχαίνουν τους δοθέντες από το παιχνίδι στόχους. Τα δύσκολα παιχνίδια εξερεύνησης, ταξινομούνται με βάση τη πυκνότητα του σήματος εξερεύνησης.

Easy Exploration			Hard exploration	
Human Optimal		Exploit	Dense Reward	Sparse Reward
ASSAULT	ASTERIX	BEAM RIDER	ALIEN	FREEWAY
ASTEROIDS	ATLANTIS	KANGAROO	AMIDAR	GRAVITAR
BATTLE ZONE	BERZERK	KRULL	BANK HEIST	MONTEZUMA'S REVENGE
BOWLING	BOXING	KUNG-FU MASTER	FROSTBITE	PITFALL!
BREAKOUT	CENTIPEDE	ROAD RUNNER	H.E.R.O	PRIVATE EYE
CHOPPER CMD	CRAZY CLIMBER	SEAQUEST	MS. PAC-MAN	SOLARIS
DEFENDER	DEMON ATTACK	UP N DOWN	Q*BERT	VENTURE
DOUBLE DUNK	ENDURO	TUTANKHAM	SURROUND	
FISHING DERBY	GOPHER	WIZARD OF WOR		
ICE HOCKEY	JAMES BOND	ZAXXON		
NAME THIS GAME	PHOENIX			
PONG	RIVER RAID			
ROBOTANK	SKIING			
SPACE INVADERS	STARGUNNER			

Πίνακας 5.1: Ταξινόμηση Περιβαλλόντων Atari(Πηγή: [1])

Τα πειράματα έγιναν για τα περιβάλλοντα Pong και MsPacman. Το Pong αποτελεί ένα από τα πιο εύκολα παιχνίδια του Gym. Ο παίκτης καλείται να ελέγξει ένα paddle και να χτυπήσει τη μπάλα ώστε να πετύχει ένα πόντο στο αντίπαλο τέρμα. Κάθε φορά που που πετυχαίνει πόντο ανταμείβεται με score 1, ενώ κάθε φορά που δέχεται πόντο ανταμείβεται με -1.

Στο MsPacman, ο παίκτης ελέγχει ένα χαρακτήρα σε ένα χάρτη λαβύρινθο, διάσπαρτο με ανταμοιβές τις οποίες καλείται να συλλέξει. Ταυτόχρονα, καλείται να αποφύγει κινδύνους-φαντάσματα σε κάποιες φάσεις του παιχνιδιού ή να τα κυνηγήσει σε άλλες φάσεις με σκοπό να πετύχει ένα πολύ υψηλό score. Η πίστα ολοκληρώνεται όταν όλες οι ανταμοιβές έχουν συλλεχθεί ανεξαρτήτως των φαντασμάτων. Περιμένουμε, ότι ένας πράκτορας με στόχο την μεγιστοποίηση του score, θα προσπαθεί να κυνηγήσει όσα περισσότερα φαντασματάκια μπορεί και καλείται να πάρει πιο υψηλού επιπέδου αποφάσεις, όπως να ρισκάρει να θυσιάσει μια ζωή

για να επιτύχει μια υψηλή ανταμοιβή. Επίσης, παρά το γεγονός ότι στην αρχή μιας πίστας το σήμα ανταμοιβής είναι πυκνό, όσο το παιχνίδι προχωράει αυτό γίνεται λιγότερο συχνό με αποτέλεσμα να περνούν πολλά βήματα μέχρι ο πράκτορας να ΄σκοντάψει΄ πάνω σε κάποια ανταμοιβή. Για αυτό συγκαταλέγεται στα δύσκολα παιχνίδια αναφορικά με την εξερεύνηση.

5.2 Λεπτομέρειες προεπεξεργασίας και Αρχιτεκτονικής Συνελικτικών επιπέδων

Τα βήματα προεπεξεργασίας των δεδομένων παραμένουν ίδια για όλους τους αλγορίθμους και περιβάλλοντα(Πίνακας 5.2). Το Gym παρέχει μια RGB εικόνα 210×160 pixel, που αντιστοιχεί στην εικόνα που βλέπει ο παίκτης του παιχνιδιού. Η εικόνα μετατρέπεται σε γκρίζα και υπόκειται σε rescaling και κανονικοποίηση με αποτέλεσμα οι εικόνες να είναι 84×84 pixel με τιμές έντασης pixel στο διάστημα $[0, 1]$.

Για να προσομοιωθεί ο ρυθμός αποφάσεων ενός ανθρώπου, αγνοούνται 3 frames κάθε 4 διαδοχικά frames, το οποίο αντιστοιχεί σε λήψη αποφάσεων με ρυθμό $15Hz$. Αυτή η διαδικασία ονομάζεται frame skip . Οι αποφάσεις που εκδίδονται από τον πράκτορα για το frame που βλέπει, επαναλαμβάνονται για τα υπόλοιπα 3 frames.

Σε κάθε βήμα ακολουθούμε την δράση που εκδίδει ο πράκτορας με πιθανότητα 25% ή την προηγούμενη δράση του πράκτορα με πιθανότητα 75%. Αυτή η διαδικασία ονομάζεται sticky actions και είναι αναγκαία για την εισαγωγή στοχαστικότητας στο περιβάλλον, καθώς τα παιχνίδια Atari είναι εκ φύσεως ντετερμινιστικά περιβάλλοντα, και αποθαρρύνει την απομνημόνευση σειράς δράσεων. Για τον ίδιο λόγο, στην αρχή κάθε επεισοδίου, αγνοούμε το πολύ τα 10 πρώτα frames, ώστε το κάθε επεισόδιο να ξεκινά με διαφορετική αρχική κατάσταση.

Επίσης, για να αποθαρρύνουμε στάσιμες συμπεριφορές του πράκτορα, εφαρμόζουμε τερματισμό του επεισοδίου και reset μετά από 4500 βήματα. Τέλος, δεν θεωρούμε το χάσιμο ζωής ως λήξη του επεισοδίου και κλιπάρουμε τις ανταμοιβές στις τιμές -1, 0, 1.

Υπερπαράμετροι	Τιμές
Grayscale	True
Μέγεθος frames	(84,84)
Frame Skip	4
Frame Stack	4
Sticky Actions	True
Max init. no-ops	10 steps
Terminal on life loss	False
Max steps per episode	4500 steps
Reward clip	-1, 0, 1

Πίνακας 5.2: Παράμετροι Προεπεξεργασίας Περιβάλλοντος

Τα frames που προκύπτουν, συλλέγονται(frame stack) σε έναν όγκο $84 \times 84 \times 4$ που αποτελεί την κατάσταση που βλέπει ο πράκτορας. Ο όγκος τίθεται ως είσοδος σε μια σειρά

από εκπαιδεύσιμα συνελικτικά επίπεδα που παράγουν μια αναπαράσταση της κατάστασης. Τα συνελικτικά επίπεδα έχουν την ίδια αρχιτεκτονική για όλους τους αλγορίθμους. Το πρώτο επίπεδο αποτελείται από 32 φίλτρα μεγέθους 8×8 με stride 4 και συνάρτηση ενεργοποίησης ReLU. Το δεύτερο επίπεδο αποτελείται από 64 φίλτρα μεγέθους 4×4 με stride 2 και συνάρτηση ενεργοποίησης ReLU. Τέλος, το τρίτο επίπεδο αποτελείται από 64 φίλτρα μεγέθους 3×3 με stride 1 και συνάρτηση ενεργοποίησης ReLU. Οι έξοδοι του τρίτου επιπέδου τίθενται ως είσοδοι σε ένα απλό επίπεδο 512 νευρώνων με συνάρτηση ενεργοποίησης ReLU, που παράγει την αναπαράσταση χαρακτηριστικών των καταστάσεων. Δεν χρησιμοποιούμε pooling επίπεδα ανάμεσα στα συνελικτικά καθώς μας ενδιαφέρει η ακριβής θέση εμφάνισης των χαρακτηριστικών στην εικόνα.

5.3 Περιγραφή αλγορίθμων

double DQN με αρχ/κή μονομαχίας και Μνήμη Εμπειριών Προτεραιότητας

Για την υλοποίηση του double DQN με αρχ/κή μονομαχίας και Μνήμη Εμπειριών Προτεραιότητας χρησιμοποιούμε τις συνελικτικές αναπαραστάσεις που περιγράφηκαν προηγούμενα. Τα features τίθενται ως είσοδοι σε ένα επίπεδο νευρώνων με γραμμική ενεργοποίηση, με μία έξοδο για τον υπολογισμό της συνάρτησης αξίας κατάστασης και μία έξοδο πλεονεκτήματος για κάθε δυνατή δράση. Οι έξοδοι αυτοι συνδυάζονται για την παραγωγή της συνάρτησης αξίας κατάστασης-δράσης σύμφωνα με την (4.27). Οι παράμετροι των κρυφών επιπέδων αρχικοποιούνται με κλίμακα 2.0 (`tf.keras.initializers.VarianceScaling` στο Tensorflow).

Οι υπερπαράμετροι που χρησιμοποιήθηκαν παρουσιάζονται στον πίνακα 5.3. Σημειώνεται ότι χρησιμοποιείται η συνάρτηση κόστους Huber [12] αντί της $L2$-νόρμας του λάθους, η οποία περιορίζει τις κλίσεις να αυξάνονται γραμμικά για μεγάλα λάθη. Οι λοιπές λεπτομέρειες του αλγορίθμου μένουν όπως παρουσιάστηκαν στις ενότητες 4.1.3. και 4.1.4 και στον αλγόριθμο 10.

A2C

Για την υλοποίηση του A2C χρησιμοποιούμε τις συνελικτικές αναπαραστάσεις που περιγράφηκαν προηγούμενα. Τα features τίθενται ως είσοδοι σε ένα επίπεδο νευρώνων με γραμμική ενεργοποίηση, με μία έξοδο για τον υπολογισμό της συνάρτησης αξίας κατάστασης και μία έξοδο πλεονεκτήματος για κάθε δυνατή δράση. Για τις παραμέτρους όλων των κρυφών επιπέδων, χρησιμοποιείται ορθογώνια αρχικοποίηση με κλίμακα $\sqrt{2}$, ενώ για τα επίπεδα εξόδου η κλίμακα είναι 0.1.

Οι υπερπαράμετροι που χρησιμοποιήθηκαν παρουσιάζονται στον πίνακα 5.4. Χρησιμοποιούμε 8 παράλληλα περιβάλλοντα, τα οποία δειγματοληπτούν την πολιτική για 5 βήματα, και στέλνουν το rollout τους στον κεντρικό πράκτορα για την παραγωγή του batch και την ανανέωση του νευρωνικού δικτύου. Ο υπολογισμός των στόχων για τον δράστη και τον κριτή γίνεται με GAE (4.45), (4.46) για κάθε rollout ξεχωριστά. Τα δεδομένα συγκεντρώνονται σε

Υπερπαράμετροι	Τιμές
Μέγεθος Batch	32
Μέγεθος Replay Memory Buffer	1000000
Μέγεθος αρχικοποίησης μνήμης	50000
Συχνότητα ανανέωσης Q	κάθε 4 βήματα
Συχνότητα ανανέωσης target	κάθε 10K βήματα
Παράγοντας έκπτωσης γ	0.99
Ρυθμός Μάθησης Q-network	0.00025/4
Αρχικό ε	1
Τελικό ε	0.1
Γραμμική μείωση ε	εντός 1M βημάτων
ε αξιολόγησης	0.05
Εκθέτης α	0.6
Εκθέτης β	από 0.4 σε 0.52 γραμμικά

Πίνακας 5.3: Υπερπαράμετροι ddDQN με PER (προσαρμογή από [8])

Υπερπαράμετροι	Τιμές
Μέγεθος Rollout	5
Αριθμός Περιβαλλόντων	8
Συντελεστής κόστους κριτή	0.5
Συντελεστής κόστους δράστη	1
Συντελεστής κόστους εντροπίας	0.01
GAE λ	0.95
Παράγοντας έκπτωσης γ	0.99
Ρυθμός Μάθησης	0.0001

Πίνακας 5.4: Υπερπαράμετροι A2C (προσαρμογή από [11])

ένα ενιαίο batch πάνω στο οποίο εκπαιδεύεται το δίκτυο δράστη-κριτή, όπως στον αλγόριθμο 12.

PPO

Ο δράστης-κριτής στον PPO ακολουθεί την ίδια αρχιτεκτονική με το A2C. Η μόνη διαφορά έγκειται στο κόστος του δράστη και στο ότι τα updates γίνονται για 4 εποχές, χωρίζοντας το batch σε 4 μικρότερα ανά εποχή. Οι υπερπαράμετροι που χρησιμοποιήθηκαν παρουσιάζονται στον πίνακα 5.5 και ο αλγόριθμος 13.

ICM

Η αρχιτεκτονική του δικτύου δράστη-κριτή παραμένει ίδια. Αναφορικά με το δίκτυο παραγωγής της ανταμοιβής περιέργειας, αυτό χρησιμοποιεί όμοια συνελικτικά επίπεδα για την

Υπερπαράμετροι	Τιμές
Μέγεθος Rollout	128
Αριθμός Περιβαλλόντων	8
Αριθμός Εποχών	4
Αριθμός mini-batch ανά εποχή	4
Συντελεστής κόστους κριτή	0.5
Συντελεστής κόστους δράστη	1
Συντελεστής κόστους εντροπίας	0.01
Παράγοντας έκπτωσης γ	0.99
Κλιπάρισμα λόγου πολιτικών	[1-0.2, 1+0.2]
Ρυθμός Μάθησης	0.0001

Πίνακας 5.5: Υπερπαράμετροι PPO (προσαρμογή από [20])

παραγωγή της αναπαράστασης κατάστασης, αλλά το τελικό πλήρως συνδεδεμένο επίπεδο που παράγει την αναπαράσταση έχει γραμμική συνάρτηση ενεργοποίησης. Για τις παραμέτρους όλων των επιπέδων, χρησιμοποιείται ορθογώνια αρχικοποίηση με κλίμακα $\sqrt{2}$

Υπερπαράμετροι	Τιμές
Μέγεθος Rollout	128
Αριθμός Περιβαλλόντων	8
Αριθμός Εποχών	4
Αριθμός mini-batch ανά εποχή	4
Αρχ/ση Παρατηρήσεων	10 rollout/περιβάλλον
Συντελεστής κόστους κριτή	0.5
Συντελεστής κόστους δράστη	1
Συντελεστής κόστους εντροπίας	0.01
Παράγοντας έκπτωσης γ	0.99
Κλιπάρισμα λόγου πολιτικών	[1-0.2, 1+0.2]
Ρυθμός Μάθησης Δράστη-Κριτή	0.0001
Ρυθμός Μάθησης ICM	0.0001
Συντελεστής Κόστους forward	0.2
Συντελεστής Κόστους inverse	0.8
Συντελεστής Στάθμισης r^{int}	0.05

Πίνακας 5.6: Υπερπαράμετροι ICM (προσαρμογή από [16])

Μια ακόμη σημαντική διαφορά είναι ότι οι είσοδοι του δικτύου κανονικοποιούνται ώστε να έχουν μέση τιμή 0 και τυπική απόκλιση 1, αντί της κανονικοποίησης στο διάστημα $[0, 1]$. Αυτό επιτυγχάνεται διατηρώντας ένα τρέχον μέσο όρο και τρέχουσα διακύμανση για τις καταστάσεις. Τα τρέχοντα αυτά στατιστικά αρχικοποιούνται, τρέχοντας ένα τυχαίο πράκτορα στην αρχή της εκπαίδευσης για 10 rollout/περιβάλλον. Η επιλογή αυτή εξασφαλίζει ότι οι αναπαραστάσεις

θα είναι σε θέση να ξεχωρίζουν μικρές λεπτομέρειες στο επίπεδο της εικόνας, αποφεύγοντας το φαινόμενο να δίνουν την ίδια έξοδο ανεξαρτήτως του όγκου εισόδου.

Το forward μοντέλο δέχεται ως εισόδους την αναπαράσταση της κατάστασης S_t και την δράση που επιλέχθηκε σε μορφή one-hot a_t και προβλέπει μια αναπαράσταση για την επόμενη κατάσταση S_{t+1} μέσω δύο διαδοχικών επιπέδων 512 νευρώνων με το κρυφό επίπεδο να έχει συνάρτηση ενεργοποίησης ReLU και το τελικό επίπεδο να είναι γραμμικό. Για τις παραμέτρους όλων των επιπέδων, χρησιμοποιείται ορθογώνια αρχικοποίηση με κλίμακα $\sqrt{2}$. Η $L2$ νόρμα του λάθους μεταξύ της προβλεπόμενης αναπαράστασης κατάστασης και της πραγματικής αναπαράστασης χρησιμοποιείται για την παραγωγή της ανταμοιβής περιέργειας.

Ένα μη επιθυμητό αποτέλεσμα της εκπαίδευσης του δικτύου παραγωγής του motivation είναι ότι όσο προχωράει η εκπαίδευση, η intrinsic ανταμοιβή θα γίνεται μικρότερη, καθώς το δίκτυο θα είναι σε θέση να προβλέψει με μεγάλη ακρίβεια τα στατικά αναλλοίωτα στοιχεία στο επίπεδο της εικόνας, όπως τους τοίχους ενός δωματίου. Για να διατηρηθεί μια συνέπεια στις ανταμοιβές καθόλη τη διάρκεια της εκπαίδευσης, διαιρούμε τις ανταμοιβές με μια τρέχουσα εκτίμηση της τυπικής απόκλισης της intrinsic επιστροφής.

Τελικά, η συνάρτηση κόστους σχηματίζεται με σκοπό την ελαχιστοποίηση της ανταμοιβής περιέργειας.

Αναφορικά με το inverse μοντέλο, αυτό δέχεται ως εισόδους τις πραγματικές αναπαραστάσεις του καταστάσεων S_t, S_{t+1}, τις επεξεργάζεται μέσω 2 διαδοχικών επιπέδων 512 νευρώνων, και παράγει μια κατανομή πιθανότητας που αντιστοιχεί στην πρόβλεψη της δράσης που έφερε την μετάβαση $S_t \rightarrow S_{t+1}$. Για τις παραμέτρους του κρυφού επιπέδου, χρησιμοποιείται ορθογώνια αρχικοποίηση με κλίμακα $\sqrt{2}$, ενώ για το επίπεδο εξόδου η κλίμακα είναι 0.1. Ως κόστος χρησιμοποιείται η ετεροεντροπία μεταξύ της προβλεπόμενης κατανομή a και της πραγματικής δράσης που έλαβε χώρα.

Τέλος, τα δύο κόστη σταθμίζονται με σταθερές υπερπαραμέτρους και οι παράμετροι της αναπαράστασης, του εμπρόσθιου και του οπίσθιου δικτύου βελτιστοποιούνται από κοινού.

RND

Το δίκτυο δράστη-κριτή παραμένει ίδιο. Το δίκτυο προβλέπτη χρησιμοποιεί την ίδια αρχιτεκτονική συνελικτικών επιπέδων. Οι αναπαραστάσεις, τίθενται ως είσοδοι σε ένα επίπεδο 512 νευρώνων με γραμμική συνάρτηση ενεργοποίησης που παράγει τη πρόβλεψη. Το τυχαία αρχικοποιημένο δίκτυο στόχος έχει την ίδια συνελικτική αρχιτεκτονική και οι έξοδοί του αποτελούνται από την αναπαράσταση. Και στα δύο δίκτυα χρησιμοποιείται ορθογώνια αρχικοποίηση με κλίμακα $\sqrt{2}$.

Ακολουθείται η ίδια διαδικασία κανονικοποίησης των παρατηρήσεων και των ανταμοιβών περιέργειας με το δίκτυο ICM.

5.4 Αποτελέσματα

Όλοι οι πράκτορες εκπαιδεύτηκαν για 10M βήματα. Οι καμπύλες εκπαίδευσης για τους διάφορους αλγορίθμους για το Pong και το MsPacman φαίνονται στα σχήματα 5.1 και 5.2.

Υπερπαράμετροι	Τιμές
Μέγεθος Rollout	128
Αριθμός Περιβαλλόντων	8
Αριθμός Εποχών	4
Αριθμός mini-batch ανά εποχή	4
Αρχ/ση Παρατηρήσεων	10 rollout/περιβάλλον
Συντελεστής κόστους κριτή	0.5
Συντελεστής κόστους δράστη	1
Συντελεστής κόστους εντροπίας	0.01
Παράγοντας έκπτωσης γ	0.99
Κλιπάρισμα λόγου πολιτικών	[1-0.2, 1+0.2]
Ρυθμός Μάθησης Δράστη-Κριτή	0.0001
Ρυθμός Μάθησης RND	0.0001
Συντελεστής Στάθμισης r^{int}	0.1

Πίνακας 5.7: Υπερπαράμετροι RND (προσαρμογή από [4])

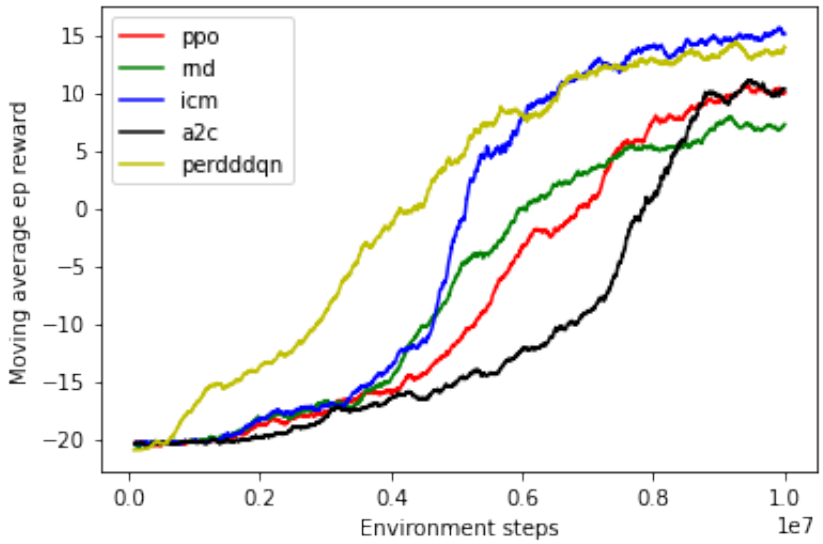
Οι καμπύλες έχουν υποστεί εξομάλυνση με κινούμενο φίλτρο μέσου όρου για 100 επεισόδια, για να είναι πιο ευανάγνωστες.

Τα τελικά αποτελέσματα φαίνονται στον πίνακα 5.8 και προκύπτουν από το εκάστοτε καλύτερο δυνατό δίκτυο κατά την διάρκεια της εκπαίδευσης. Για την εξαγωγή του καλύτερου δικτύου, παγώνουμε την διαδικασία της εκπαίδευσης κάθε 1M βήματα και αξιολογούμε τους πράκτορες για 30 επεισόδια. Οι παράμετροι των δικτύων που πετυχαίνουν το καλύτερο μέσο score αποθηκεύονται. Η διαδικασία αυτή προσομοιάζει τον έγκαιρο τερματισμό(early stop) που χρησιμοποιείται συχνά στην μηχανική μάθηση με εκπαιδευτή. Στο τέλος της εκπαίδευσης αξιολογούμε εκ νέου το καλύτερο δίκτυο που έχει προκύψει για 100 επεισόδια και αυτό είναι το μέσο score που παρουσιάζουμε στον πίνακα 5.8.

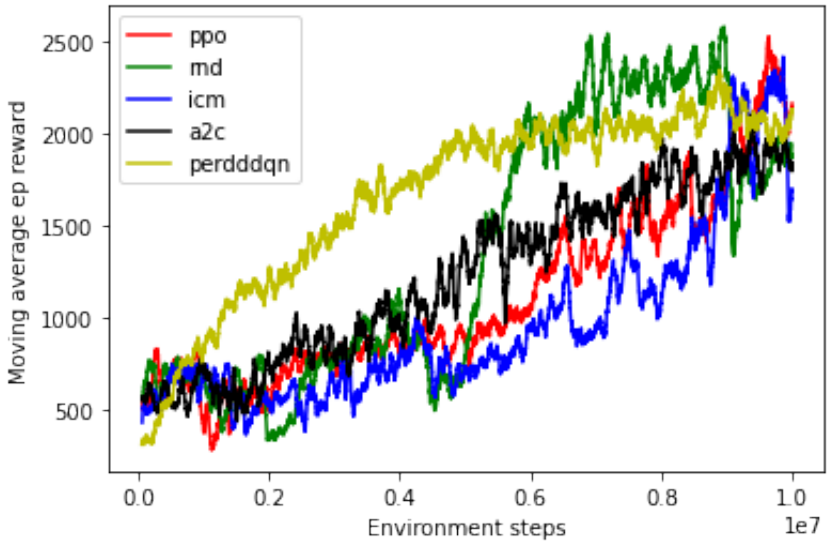
Αλγόριθμος	Pong	MsPacman
ε-greedy ddDQN(PER)	13.94	2128.6
A2C	10.29	1844.2
PPO	9.99	2165.7
RND	7.25	**2370.1**
ICM	**15.94**	2139.0

Πίνακας 5.8: Αποτελέσματα 100 επεισοδίων μετά από εκπαίδευση για 10M βήματα

Παρατηρούμε ότι οι μέθοδοι που βασίζονται στη περιέργεια πετυχαίνουν καλύτερα score από τις μεθόδους που χρησιμοποιούν ε-άπληστες πολιτικές ή κόστος εντροπίας ως τον μόνο μηχανισμό εξερεύνησης. Συγκεκριμένα, το ICM πετυχαίνει το καλύτερο score στο περιβάλλον Pong, ενώ το RND το καλύτερο score στο MsPacman. Παρόλα αυτά το μικρό μέγεθος των πειραμάτων, καθώς και η μικρές διαφορές στα αποτελέσματα δεν είναι αρκετές για να εξάγουμε



Σχήμα 5.1: Καμπύλες εκπαίδευσης για το Pong(μέσος όρος score για 100 επεισόδια)

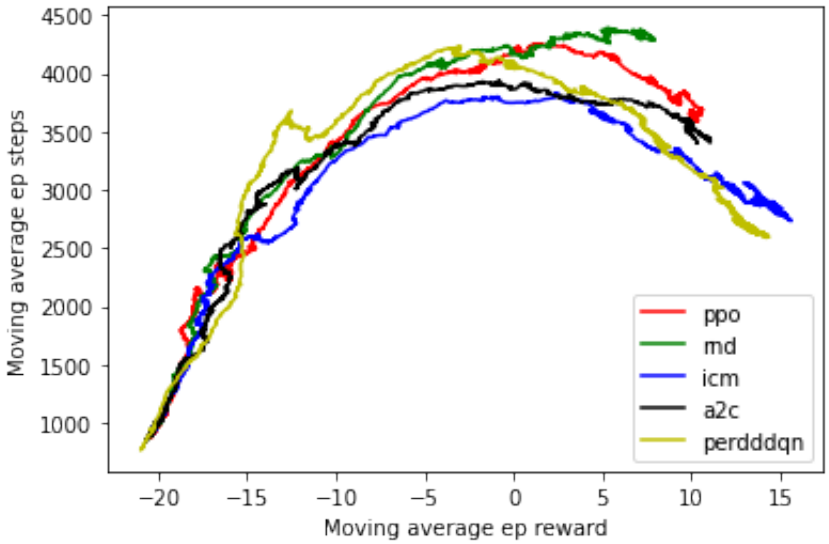


Σχήμα 5.2: Καμπύλες εκπαίδευσης για το MsPacman(μέσος όρος score για 100 επεισόδια)

ασφαλή συμπεράσματα ως προς την ποιότητα των πρακτόρων αναφορικά με την εξερεύνηση.

Είναι εμφανές ότι το double DQN με μνήμη εμπειριών προτεραιότητας πετυχαίνει καλό μέσο όρο score πολύ νωρίτερα από ότι η μέθοδοι που βασίζονται στην πολιτική και επίσης φαίνεται να είναι λιγότερο ευάλωτο σε ακραίες μεταβολές του μέσου score. Παρόλα αυτά παρατηρείται ότι μένει στάσιμο μετά από κάποιο αριθμό επαναλήψεων, σε αντίθεση με τους αλγορίθμου κλίσης πολιτικής που δεν έχουν φτάσει το πλατώ τους μετά από εκπαίδευση 10M βημάτων.

Η έλλειψη σημαντικών βελτιώσεων των δύο μεθόδων περιέργειας, σε σύγκριση με τις ε-άπληστες πολιτικές του DQN, μπορεί εν μέρει να οφείλεται στην μνήμη προτεραιότητας. Μια



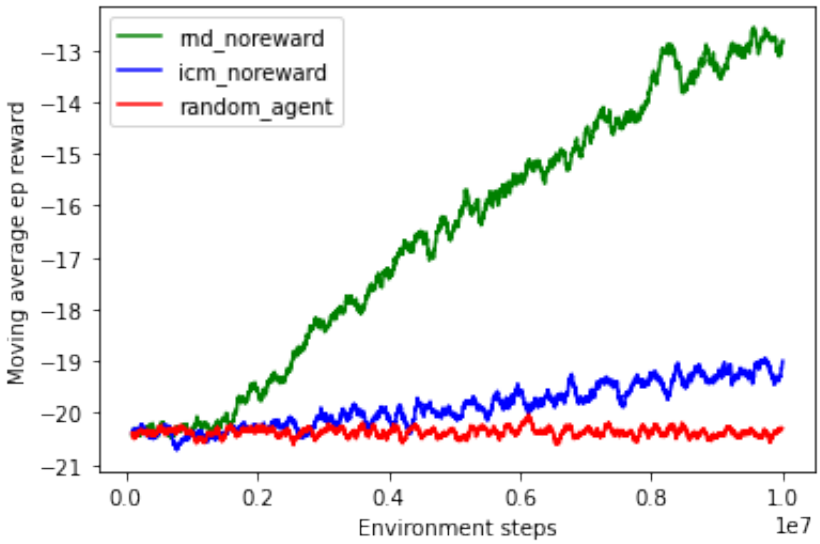
Σχήμα 5.3: Μέσος αριθμός βημάτων για την επίτευξη μέσου score στο Pong

σημαντική εμπειρία στα πλαίσια της μνήμης προτεραιότητας, μπορεί να παρατηρείται σπάνια, αλλά δειγματοληπτείται για εκπαίδευση πολύ συχνότερα. Έτσι, ο πράκτορας μπορεί να μαθαίνει από την εμπειρία αυτή χωρίς την ανάγκη να την επισκέπτεται συχνά με εξερεύνηση.

Η σύγκριση των μεθόδων περιέργειας, με τις βασικές μεθόδους κλίσης πολιτικής είναι πιο ευθεία. Το σήμα περιέργειας που παράγεται από τον ICM και RND είναι μία τάξη μεγέθους μικρότερο σε σχέση με το σήμα ανταμοιβής που παράγει το περιβάλλον και έτσι υπό την παρουσία εξωτερικών σημάτων ανταμοιβής σε ένα rollout είναι σχεδόν αμελητέο. Παρόλα αυτά, το γεγονός ότι ακόμα και στα περιβάλλοντα όπως το Pong και MsPacman, που θεωρούνται περιβάλλοντα πυκνών ανταμοιβών, το σήμα περιέργειας είναι πολύ συχνότερο από το σήμα ανταμοιβής, σημαίνει ότι οι αντίστοιχες επιστροφές και τα πλεονεκτήματα είναι συγκρίσιμα και στην ίδια τάξη μεγέθους. Αυτό είναι σημαντικό γιατί οι επιστροφές και τα πλεονεκτήματα είναι τα δεδομένα που χρησιμοποιούνται ως στόχοι στην εκπαίδευση του δράστη-κριτή. Για παράδειγμα, στο Pong μπορεί να υπάρξουν πολλά διαδοχικά rollout στα οποία ο πράκτορας δεν λαμβάνει κανένα σήμα ανταμοιβής από το περιβάλλον. Σε αυτές τις περιπτώσεις, η πολιτική εκπαιδεύεται αποκλειστικά από τις επιστροφές και τα πλεονεκτήματα περιέργειας.

Στο σχήμα 5.3, φαίνεται ο μέσος αριθμός βημάτων του πράκτορα για την επίτευξη ενός μέσου score για το Pong. Παρατηρούμε ότι, κατά μέσο όρο το RND χρειάζεται 500-1000 περισσότερα βήματα ανά επεισόδιο για την επίτευξη του ίδιου score με τους υπόλοιπους αλγορίθμους. Αυτό μπορεί να αποδοθεί σε περισσότερη εξερεύνηση αλλά σε ένα περιβάλλον που η περισσότερη εξερεύνηση δεν αποδίδει καλύτερα αποτελέσματα. Από εξέταση των video παιχνιδιού κατά τη διάρκεια της εκπαίδευσης, παρατηρήθηκε ότι, το RND στο Pong επιλέγει να επεκτείνει τη διάρκεια του παιχνιδιού, επιχειρώντας μεγάλης διάρκειας γύρους, κατά πάσα πιθανότητα οδηγούμενο από την περιέργεια. Παρόμοιο φαινόμενο δεν παρατηρείται στον ICM.

Όσον αφορά το MsPacman σχήματα όπως το 5.3 δεν μπορούν να χρησιμοποιηθούν για την εξαγωγή συμπερασμάτων, καθώς στο περιβάλλον αυτό παρατηρείται συχνά αδράνεια του



Σχήμα 5.4: Μέσο score στο Pong μόνο με σήμα περιέργειας

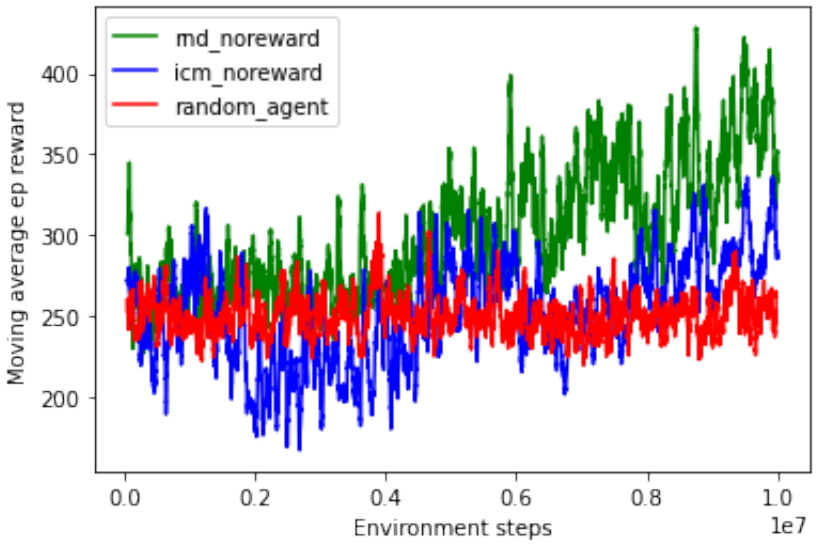
παίκτη, η οποία μπορεί λανθασμένα να ερμηνευτεί ως εξερεύνηση. Η εξέταση των video παιχνιδιού κατά τη διάρκεια της εκπαίδευσης δεν δείχνει οι πράκτορες περιέργειας να επιδίδονται σε καλύτερες στρατηγικές εξερεύνησης, ειδικά στα τελικά στάδια του παιχνιδιού όπου οι ανταμοιβές του περιβάλλοντος έχουν λιγοστέψει.

Τέλος, εκπαιδεύουμε ένα πράκτορα PPO δίνοντας του μόνο το σήμα ανταμοιβής περιέργειας, και καθόλου εξωτερικό σήμα ανταμοιβής, για να διαπιστώσουμε το κατά πόσο η περιέργεια συνεισφέρει στην εξερεύνηση $r_{total} = r^{int}$. Αναμένουμε οι πράκτορες να επιτύχουν score παρά το γεγονός ότι δεν είναι αυτός ο σκοπός τους, καθώς τα παιχνίδια Atari είναι σχεδιασμένα έτσι ώστε να δίνουν παθητικά score σε πράκτορες που απλά εξερευνούν το χώρο καταστάσεων. Τα αποτελέσματα φαίνονται στα σχήματα 5.4, 5.5.

Παρατηρούμε ότι ο πράκτορας που εκπαιδεύεται με βάση το σήμα ανταμοιβής περιέργειας του RND πετυχαίνει εμφανώς καλύτερα αποτελέσματα από ένα τυχαίο πράκτορα. Το ίδιο μπορεί να ειπωθεί και για το ICM παρά το γεγονός ότι τα κέρδη είναι πολύ μικρότερα. Αυτό συμβαίνει κυρίως λόγω της μη ρύθμισης των υπερπαραμέτρων και της αρχιτεκτονικής του ICM. Από τα πειράματα αυτά καταδεικνύεται ότι οι τεχνικές εξερεύνησης βασισμένες στη περιέργεια μπορούν να επιτύχουν μακροβιότερη εξερεύνηση σε σχέση με εξερεύνηση που βασίζεται στην τύχη.

5.5 Συμπεράσματα και μελλοντικές κατευθύνσεις

Συμπερασματικά, η χρήση μηχανισμών περιέργειας σε περιβάλλοντα με πυκνό σήμα ανταμοιβής, φαίνεται να μην βελτιώνει την εξερεύνηση σημαντικά. Τα πειράματα που εξετάστηκαν βέβαια απέχουν πολύ από τη κλίμακα στα αντίστοιχα πειράματα όπως στο [16], [4], [5]. Μια βασική διαφορά είναι ο χρόνος εκπαίδευσης που συνήθως είναι της τάξης των 100M βημάτων, και απαιτεί υπολογιστικούς πόρους που δεν ήταν διαθέσιμοι για την εκπόνηση αυτής της εργασίας.



Σχήμα 5.5: Μέσο score στο MsPacman μόνο με σήμα περιέργειας

Επίσης, η διαδικασία ρύθμισης των υπερπαραμέτρων ειδικά για τις μεθόδους περιέργειας διαφέρει, καθώς οι αλγόριθμοι περιέργειας συνηθίζεται να ρυθμίζονται με βάση περιβάλλοντα αραιών ανταμοιβών όπως το MontezumasRevenge. Αυτή η πρακτική αποθαρρύνεται στο [25], καθώς οδηγεί σε πλασματικά αποτελέσματα σχετικά με το επίπεδο εξερεύνησης που επιτυγχάνεται από τις διάφορες μεθόδους περιέργειας. Μάλιστα, στο [25], εξάγονται συμπεράσματα παρόμοια με αυτά που παρουσιάστηκαν, σχετικά με τη χρήση μηχανισμών περιέργειας στο σύνολο των περιβαλλόντων που παρέχονται από τα παιχνίδια Atari.

Τέλος, η μεγάλη δυσκολία στη ρύθμιση υπερπαραμέτρων, ειδικά στο ICM, υπογραμμίζει μια γενικότερη κριτική προς την ενισχυτική μάθηση που αφορά την υπερβολική εξάρτηση των αποτελεσμάτων από τις υπερπαραμέτρους. Αυτό, σε συνδυασμό με τον μεγάλο χρόνο που χρειάζονται τα πειράματα για να παράξουν αποτελέσματα, έχει ως συνέπεια ο κύκλος μεταξύ δοκιμών να διαρκεί εβδομάδες ακόμα και σε απλά περιβάλλοντα όπως το Atari.

Ως εκ τούτου, η ανάπτυξη ενός αποδοτικού μηχανισμού ρύθμισης υπερπαραμέτρων αποτελεί μια σημαντική μελλοντική κατεύθυνση, καθώς τεχνικές όπως το grid-search, random search και fine-tuning είναι ιδιαίτερα κοστοβόρες σε χρόνο υπολογισμού.

Μια ακόμη μελλοντική κατεύθυνση της παρούσας εργασίας, αφορά την βελτίωση της χρησιμοποίησης δεδομένων, ειδικά για τους αλγορίθμους κλίσης πολιτικής. Το γεγονός ότι χρειάζονται δεδομένα εκπαίδευσης που αντιστοιχούν σε μήνες παιχνιδιού σε πραγματικό χρόνο για την εκπαίδευση πρακτόρων, μας κάνει να θεωρούμε ότι υπάρχουν αποδοτικότεροι τρόποι χρησιμοποίησης των δεδομένων που συλλέγει ο πράκτορας. Η χρήση μνήμης εμπειριών από πράκτορες που βασίζονται στο θεώρημα κλίσης πολιτικής αποτελεί μια πρώτη κατεύθυνση. Ακόμη, η χρήση transfer learning ενδέχεται να αποφέρει μείωση του χρόνου εκπαίδευσης και αξίζει να ερευνηθεί.

Αναφορικά με τους μηχανισμούς παραγωγής περιέργειας, όπως είδαμε, αυτοί χωρίζονται σε δύο βασικές κατηγορίες: την ανίχνευση καινοτόμων καταστάσεων(RND) και τη δυνατότητα

πρόβλεψης της δυναμικής του περιβάλλοντος(ICM). Κανένας από αυτούς τους μηχανισμούς δεν λαμβάνει υπόψη τους στόχους του περιβάλλοντος και άρα δεν είναι σε θέση να διαφοροποιήσει ΄περίεργες΄ καταστάσεις από τις πραγματικά χρήσιμες. Αυτό θεωρητικά επαφύεται στον πράκτορα(για παράδειγμα τον δράστη-κριτή), αλλά ο βομβαρδισμός από ανταμοιβές περιέργειας μπορεί να αποσπάσει τη προσοχή του και να έχει ως αποτέλεσμα την χρήση ατελέσφορων πολιτικών. Μια κατεύθυνση αφορά τη ροή πληροφορίας από τον πράκτορα προς το σύστημα που παράγει το σήμα περιέργειας με τέτοιο τρόπο ώστε, το δίκτυο περιέργειας να εκπαιδεύεται με σκοπό την ελαχιστοποίηση ενός επιπλεόν κόστους που αφορά τη γνώμη του κριτή σχετικά με τη χρησιμότητα μιας κατάστασης

Τέλος, η αρχιτεκτονική των νευρωνικών δικτύων έχει μείνει στάσιμη στο πλαίσιο των παιχνιδιών Atari, και ίσως εξελίξεις προς αυτή τη κατεύθυνση να χρήζουν εξερεύνησης. Τα αναδρομικά νευρωνικά δίκτυα μπορούν να χρησιμοποιηθούν για την αξιοποίηση της χρονικής συσχέτισης του προβλήματος επιλογής δράσεων. Επίσης, αλλαγές στην αρχιτεκτονική μπορούν να χρησιμοποιηθούν για τον συνδυασμό του εξωτερικού σήματος ανταμοιβών και του σήματος περιέργειας. Παρόλα αυτά, ακόμη και ο συνδυασμός των δύο σημάτων ανταμοιβών με χρήση διαφορετικών αρχιτεκτονικών κριτή, όπως στο [4], καταφεύγει στη αυθαίρετη στάθμιση των εξωτερικών πλεονεκτημάτων και πλεονεκτημάτων περιέργειας μέσω υπερπαραμέτρων και δεν διαφέρει πολύ από τη αυθαίρετη στάθμιση ανταμοιβών που χρησιμοποιήθηκε στην παρούσα εργασία.

Βιβλιογραφία

[1] Marc G. Bellemare et al. *Unifying Count-Based Exploration and Intrinsic Motivation*. 2016. arXiv: 1606.01868 [cs.AI].

[2] Dimitri P. Bertsekas. *Dynamic Programming and Optimal Control*. 2nd. Athena Scientific, 2000. ISBN: 1886529094.

[3] Greg Brockman et al. *OpenAI Gym*. 2016. eprint: arXiv:1606.01540.

[4] Yuri Burda et al. *Exploration by Random Network Distillation*. 2018. arXiv: 1810.12894 [cs.LG].

[5] Yuri Burda et al. *Large-Scale Study of Curiosity-Driven Learning*. 2018. arXiv: 1808.04355 [cs.LG].

[6] Hado van Hasselt, Arthur Guez, and David Silver. *Deep Reinforcement Learning with Double Q-learning*. 2015. arXiv: 1509.06461 [cs.LG].

[7] Simon S. Haykin. *Neural networks and learning machines*. Third. Pearson Education, 2009.

[8] Matteo Hessel et al. *Rainbow: Combining Improvements in Deep Reinforcement Learning*. 2017. arXiv: 1710.02298 [cs.AI].

[9] Diederik P. Kingma and Jimmy Ba. *Adam: A Method for Stochastic Optimization*. 2017. arXiv: 1412.6980 [cs.LG].

[10] Y. Lecun et al. "Gradient-based learning applied to document recognition". In: *Proceedings of the IEEE* 86.11 (1998), pp. 2278–2324. DOI: 10.1109/5.726791.

[11] Volodymyr Mnih et al. *Asynchronous Methods for Deep Reinforcement Learning*. 2016. arXiv: 1602.01783 [cs.LG].

[12] Volodymyr Mnih et al. "Human-level control through deep reinforcement learning". In: *Nature* 518 (2015), pp. 529–533.

[13] Andrew Ng. *Deep Learning Specialization*. https://www.coursera.org/specializations/deep-learning. 2015.

[14] OpenAI. *OpenAI Baselines: ACKTR & A2C*. https://openai.com/blog/baselines-acktr-a2c/. 2017.

[15] OpenAI. *Reinforcement Learning with Prediction-Based Rewards*. https://openai.com/blog/reinforcement-learning-with-prediction-based-rewards/. 2018.

[16] Deepak Pathak et al. *Curiosity-driven Exploration by Self-supervised Prediction*. 2017. arXiv: 1705.05363 [cs.LG].

[17] Sebastian Ruder. *An overview of gradient descent optimization algorithms*. 2017. arXiv: 1609.04747 [cs.LG].

[18] Tom Schaul et al. *Prioritized Experience Replay*. 2016. arXiv: 1511.05952 [cs.LG].

[19] John Schulman et al. *High-Dimensional Continuous Control Using Generalized Advantage Estimation*. 2018. arXiv: 1506.02438 [cs.LG].

[20] John Schulman et al. *Proximal Policy Optimization Algorithms*. 2017. arXiv: 1707.06347 [cs.LG].

[21] David Silver. *Introduction to Reinforcement Learning*. https://deepmind.com/learning-resources/-introduction-reinforcement-learning-david-silver. 2015.

[22] Richard S Sutton and Andrew G Barto. *Reinforcement learning: An introduction*. MIT press, 2018.

[23] Richard S Sutton et al. "Policy Gradient Methods for Reinforcement Learning with Function Approximation". In: *Advances in Neural Information Processing Systems*. Ed. by S. Solla, T. Leen, and K. Müller. Vol. 12. MIT Press, 2000. URL: https://proceedings.neurips.cc/paper/1999/file/464d828b85b0bed98e80ade0a5c43b0f-Paper.pdf.

[24] Csaba Szepesvari. *Algorithms for Reinforcement Learning*. Morgan and Claypool Publishers, 2010. ISBN: 1608454924.

[25] Adrien Ali Taïga et al. *On Bonus-Based Exploration Methods in the Arcade Learning Environment*. 2021. arXiv: 2109.11052 [cs.LG].

[26] Ziyu Wang et al. *Dueling Network Architectures for Deep Reinforcement Learning*. 2016. arXiv: 1511.06581 [cs.LG].