National Technical University of Athens
School of Electrical and Computer Engineering
Division of Signals, Control and Robotics

Action to Object Knowledge Distillation for Object-centric Representation Learning

Nikolaos Giannakakis

Supervisor: P. Maragos, Professor NTUA
Co-supervisor: G. Retsinas, Postdoctoral Researcher NTUA

Computer Vision, Speech Communication & Signal Processing Group

Athens, October 2024

National Technical University of Athens
School of Electrical and Computer Engineering
Division of Signals, Control and Robotics
Computer Vision, Speech Communication & Signal Processing Group

Action to Object Knowledge Distillation for Object-centric Representation Learning

Nikolaos Giannakakis

Supervisor: P. Maragos, Professor NTUA
Co-supervisor: G. Retsinas, Postdoctoral Researcher NTUA

Approved by the Examining Committee on 24 October, 2023.

.....	.....	.....
Petros Maragos	Athanasios Rontogiannis	Ioannis Kordonis
Professor NTUA	Associate Professor NTUA	Assistant Professor NTUA

Athens, October 2024

.....

Nikolaos Giannakakis

Graduate of Electrical and Computer Engineering NTUA

Copyright © — Nikolaos Giannakakis, 2024.
This work is licensed under a Creative Commons Attribution 4.0 International License. To view a copy of this license, visit http://creativecommons.org/licenses/by/4.0/

The views and conclusions contained in this document are those of the author and should not be construed as representing the official positions of the National Technical University of Athens.

Περίληψη

Ο σκοπός αυτής της διπλωματικής εργασίας είναι η διερεύνηση της βελτίωσης της αποτελεσματικότητας των *αντικειμενοκεντρικών κωδικοποιητών εικόνας* με τεχνικές ενσωμάτωσης πληροφορίας εστιασμένης σε δράσεις. Πρώτον, δοκιμάζουμε μια αντικειμενοκεντρική μέθοδο για την απόσταξη των αναπαραστάσεων ενός προ-εκπαιδευμένου *Video Masked Auto-encoder* (Video MAE) στις αναπαραστάσεις δύο state-of-the-art κωδικοποιητών εικόνας. Η αξιολόγηση γίνεται πάνω στο πρόβλημα *κατηγοριοποίηση προσφερόμενων δυνατοτήτων αντικειμένων* (affordance categorization). Στην αξιολόγηση γίνεται χρήση ενός συνόλου δεδομένων, μικρής κλίμακας, που δημιουργήθηκε για τα πειράματα της διπλωματικής αυτής, χρησιμοποιώντας ως βάση το σύνολο δεδομένων Something-Something v2 (SSV2). Τα αποτελέσματα δείχνουν ότι οι αναπαραστάσεις του Video MAE, περιέχουν χρήσιμη πληροφορία για τους κωδικοποιητές εικόνας και δοκιμάζουμε μερικές μεθόδους για να εμπλουτίσουμε τις αναπαραστάσεις των κωδικοποιητών εικόνας. Οι μέθοδοι παρουσίασαν μια μικρή βελτίωση αλλά ίσως χρειαστούν προσαρμογές ή μεγαλύτερα σύνολα δεδομένων για την καλύτερη αξιοποίηση αυτών των αναπαραστάσεων. Επιπλέον, μελετούμε μια μέθοδο βασισμένη στην *αντικειμενοκεντρική* μέθοδο εκμάθησης αναπαραστάσεων *Slot Attention*. Η αποτελεσματικότητα της μεθόδου αξιολογείται επίσης στο πρόβλημα της κατηγοριοποίησης προσφερόμενων δυνατοτήτων και παρουσιάζει ανταγωνιστικά αποτελέσματα, ενώ επιτυγχάνει επίσης αυτόματη τμηματοποίηση των εικόνων και σημαντική μείωση του μεγέθους της αναπαράστασης ανά αντικείμενο. Τέλος, προτείνουμε μια μέθοδο για να συνδυάσουμε αντικειμενοκεντρικές αναπαραστάσεις από ένα μοντέλο βασισμένο στη μέθοδο slot attention για να παραγάγουμε μια συνολική αναπαράσταση από μια εικόνα, με στόχο την εκμάθηση *οπτικοκινητικών πολιτικών*. Αυτή η μέθοδος αξιολογείται σε μια *προσομοίωση ρομποτικού* χειρισμού και στα πειράματα που πραγματοποιήθηκαν παρουσιάζει καλύτερα αποτελέσματα σε σύγκριση με άλλες αναπαραστάσεις. Δημιουργώντας συσχετίσεις δράσης-αντικειμένου στις αναπαραστάσεις των κωδικοποιητών εικόνας, αυτή η διπλωματική επιδιώκει να συμβάλει στην ανάπτυξη πιο αποτελεσματικών συστημάτων όρασης για ρομπότ και τεχνητούς πράκτορες, επιτρέποντάς τους να κατανοούν καλύτερα τη σημασιολογία και τη δυναμική της αλληλεπίδρασης πράκτορα-αντικειμένου.

Λέξεις κλειδιά: Αντικειμενοκεντρική Εκμάθηση Αναπαραστάσεων, Vision Transformer, Masked Auto-encoder, Slot Attention, Κατηγοριοποίηση Προσφερόμενων Δυνατοτήτων Αντικειμένων, Προσομοίωση Ρομποτικού Χειρισμού.

Abstract

This thesis aims to study the possible improvement of *object-centric image encoders* by enhancing them with action-centric representations derived from videos of actions. Firstly, we study a method to distill the representations of a pre-trained *Video Masked Auto-encoder* (Video MAE) to the representations of two state-of-the-art image encoders in an object-centric manner. This method is evaluated in the task of *affordance categorization* using a small-scale dataset that we created using the Something-Something v2 (SSV2) dataset. Experiments show that the representations of the Video MAE contain information that could be useful to the image encoders, and we test some methods to enrich them with this information. The experiments show that the methods produce a marginal yet consistent enhancement. Further experimentation with larger scale model implementations and datasets could potentially unlock additional improvements. Furthermore, we propose and study a method based on the *Slot Attention* object-centric representation learning framework. The effectiveness of the method is also evaluated in the task of affordance categorization and it presents competitive results while also achieving automatic segmentation of the images and a substantial reduction in per-object representation size. Finally, we propose a method to combine object-centric representations from a slot-attention-based model to produce a flat representation vector for an image with the aim of learning *visuomotor* policies. This method is evaluated in a *robotic simulation task* and presents better results compared to other out-of-domain representations. We also show that the slot representations' performance in the simulated robotic manipulation can be improved when fine-tuning the model with videos of actions from the SSV2 dataset. By creating action-object associations in the representations of object-centric image encoders, this study seeks to contribute to the development of more effective vision perception systems for robots and artificial agents, enabling them to better understand the semantics and dynamics of agent-object interaction.

Keywords: Object-centric Representation Learning, Vision Transformer, Masked Auto-encoder, Slot Attention, Affordance Categorization, Robotic Perception, Robotics Simulation

Ευχαριστίες

Αρχικά, επιθυμώ να εκφράσω τις ευχαριστίες μου στον Καθηγητή κ. Πέτρο Μαραγκό, ο οποίος υπήρξε σημαντική πηγή γνώσης, έμπνευσης και καθοδήγησης, τόσο κατά τη διάρκεια των σπουδών μου, όσο και κατά τη διαδικασία εκπόνησης της παρούσας διπλωματικής εργασίας. Επίσης, η επιτυχής ολοκλήρωση της εργασίας αυτής οφείλεται στη συνεχή υποστήριξη και στην πολύτιμη πείρα του συν-επιβλέποντα Δρ. Γιώργου Ρετσινά, τον οποίο ευχαριστώ θερμά για τον χρόνο και την ενέργεια που μου διέθεσε σε αυτή την προσπάθεια. Τέλος, θα ήθελα να αφιερώσω την εργασία αυτή, στη σύντροφό μου Γιασμίν και στην οικογένεια μου για τη συνεχή υποστήριξη και ενθάρρυνση σε όλη τη διάρκεια αυτών των ετών.

Νίκος Γιαννακάκης
14 Οκτώβριου 2024

Contents

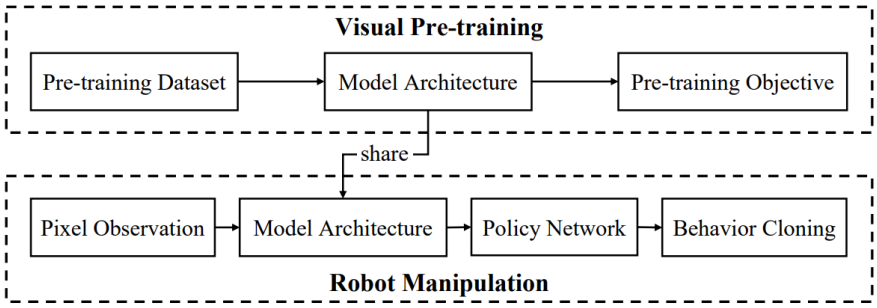
List of Figures

List of Tables

1 Εκτεταμένη Περίληψη στα Ελληνικά

1.1 Κίνητρο

Ένας βασικός στόχος του τομέα της όρασης υπολογιστών είναι η δημιουργία χρήσιμων αναπαραστάσεων και η ανάπτυξη τεχνικών για την αποτελεσματική εξαγωγή τους. Το πεδίο της ρομποτικής, έχει πολλά ανοικτά προβλήματα που μελετώνται αυτή τη στιγμή [6, 125, 2, 22, 23]. Για αρκετά από αυτά τα προβλήματα, η εξαγωγή οπτικών αναπαραστάσεων μέσω μεθόδων προ-εκπαίδευσης με οπτικά δεδομένα (Σχήμα 1) είναι πολλά υποσχόμενη, επειδή μειώνει τον χρόνο εκπαίδευσης και βελτιώνει την απόδοση και την ικανότητα γενίκευσης, σε σύγκριση με μεθόδους που εκπαιδεύουν ολόκληρο το μοντέλο από την αρχή και από άκρο σε άκρο [52, 61, 95, 128]. Αυτές οι αναπαραστάσεις θα πρέπει να μπορούν να είναι χρήσιμες σε μια ποικιλία από εργασίες και ιδανικά να απαιτούν ελάχιστη επανεκπαίδευση [95, 70].



Σχήμα 1: Οπτική προ-εκπαίδευση για ρομποτικά συστήματα. Πηγή: [52]

Μία οικογένεια μεθόδων οπτικής προ-εκπαίδευσης που παρουσιάζει αρκετό ενδιαφέρον τα τελευταία χρόνια είναι η *αντικειμενοκεντρική εκμάθηση αναπαραστάσεων*. Ο στόχος των μεθόδων αυτών είναι η αναπαράσταση σύνθετων σκηνών διαχωρίζοντάς τες σε σημασιολογικές ενότητες που ονομάζονται αντικείμενα. Αυτές οι μέθοδοι, είναι συμβατές με τον τρόπο που οι άνθρωποι επεξεργάζονται τα οπτικά σήματα οργανώνοντάς τα σε αντικείμενα [77] και παρουσιάζουν προοπτική βελτίωσης των ικανοτήτων γενίκευσης, εξηγησιμότητας και αποδοτικότητας ως προς τα δείγματα κατά την εκπαίδευση των μοντέλων [67, 8, 3].

Η αντικειμενοκεντρική εκμάθηση αναπαραστάσεων μπορεί να αντλήσει έμπνευση από τον τομέα της ψυχολογίας, όπου έχει υπάρξει εκτενής μελέτη για το πώς οι άνθρωποι μαθαίνουν να αλληλεπιδρούν με το περιβάλλον τους, συσχετίζοντας δράσεις και λέξεις με αντικείμενα. Πειράματα στην αναπτυξιακή ψυχολογία δείχνουν ότι τα βρέφη πρώτα εστιάζουν στις συσχετίσεις δράσης-αντικειμένου, ενώ οι συσχετίσεις λέξεων-αντικειμένου γίνονται σημαντικές αργότερα στην ανάπτυξή τους [24]. Οι περιορισμοί στους υπολογιστικούς πόρους, στα σύνολα δεδομένων και στις μεθοδολογίες τεχνητής εκμάθησης εμποδίζουν τα συστήματα τεχνητής νοημοσύνης να ακολουθήσουν πιστά τα στάδια της ανθρώπινης ανάπτυξης. Ωστόσο, οι τομείς αυτοί μπορούν να αποτελέσουν έμπνευση για αλγόριθμους που επιδιώκουν να προ-εκπαιδεύσουν τεχνητά συστήματα αντίληψης υποδεικνύοντας την κατεύθυνση για μάθηση τύπου curriculum learning [5, 100], η οποία αρχικά εστιάζει στην εξαγωγή αναπαραστάσεων από δράσεις και στη συνέχεια στη μάθηση βασισμένη στη γλώσσα. Η διαδικασία αυτή προχωρά από την εκμάθηση αναπαραστάσεων χαμηλού επιπέδου προς αναπαραστάσεις υψηλού επιπέδου, εστιάζοντας σταδιακά σε πιο σύνθετη και αφηρημένη πληροφορία. Επίσης, η εκμάθηση αυτή κάνει πρώτα χρήση αυτο-επιβλεπόμενων τεχνικών σε σύνολα δεδομένων που είναι πιο προσιτά, όπως μη επισημειωμένα σύνολα βίντεο και στη συνέχεια γίνεται χρήση επισημειωμένων συνόλων δεδομένων, τα οποία τείνουν να είναι μεγαλύτερου κόστους.

Αυτή η διπλωματική επικεντρώνεται σε τρόπους με τους οποίους οι δράσεις μπορούν να συσχετιστούν με αντικείμενα. Η προσπάθεια αυτή βασίζεται σε θετικά αποτελέσματα μεθόδων προεκπαίδευσης που εστιάζουν στη μοντελοποίηση δρασεοκεντρικής πληροφορίας [87, 54, 61, 73, 69] χρησιμοποιώντας σύνολα δεδομένων που καταγράφουν τον τρόπο με τον οποίο οι άνθρωποι δρουν και αλληλεπιδρούν με αντικείμενα [36, 17]. Αυτά τα σύνολα δεδομένων μπορούν να χρησιμοποιηθούν για να εκπαιδεύσουν τα συστήματα όρασης και να δώσουν στους πράκτορες ένα προβάδισμα στην κατανόηση των αλληλεπιδράσεων με αντικείμενα στον πραγματικό κόσμο.

Βάσει των παραπάνω, ο κύριος στόχος αυτής της διπλωματικής είναι να εξερευνήσει μεθόδους που μπορούν να βελτιώσουν τους αντικειμενοκεντρικούς κωδικοποιητές εικόνας, μέσω μοντελοποίησης συνόλων δεδομένων που εστιάζουν στις δράσεις και στον τρόπο αλληλεπίδρασης των ανθρώπων με αντικείμενα. Η πρώτη εργασία που χρησιμοποιείται για την αξιολόγηση της αποτελεσματικότητας αυτών των αναπαραστάσεων είναι η *κατηγοριοποίηση προσφερόμενων δυνατοτήτων αντικειμένων* (affordance categorization). Η αναγνώριση προσφερόμενων δυνατοτήτων μπορεί να βοηθήσει τα συστήματα να προβλέψουν και να σχεδιάσουν, παρέχοντας πληροφορίες για πιθανές αλληλεπιδράσεις με αντικείμενα και με το περιβάλλον. Επιπλέον, η αποτελεσματικότητα των αναπαραστάσεων αξιολογείται μέσω μιας προσομοιωμένης εργασίας ρομποτικού χειρισμού.

1.2 Συνεισφορές

Οι συνεισφορές της διπλωματικής είναι οι εξής:

1. **Something's Affordances: συλλογή ενός συνόλου δεδομένων μικρής κλίμακας για το πρόβλημα κατηγοριοποίησης δυνατοτήτων αντικειμένων.** Το Something's Affordances είναι ένα μικρής κλίμακας σύνολο δεδομένων που επεκτείνει το σύνολο δεδομένων Something-Something v2 (SSV2) [35]και εστιάζει στην κατηγοριοποίηση προσφερόμενων δυνατοτήτων αντικειμένων. Από το αρχικό σύνολο δεδομένων, επιλέχθηκε ένα μικρό υποσύνολο κατηγοριών δράσεων με βάση την ικανότητά τους να δοκιμάσουν τις μεθόδους εκμάθησης αναπαραστάσεων. Οι ετικέτες προσφερόμενων δυνατοτήτων εξήχθησαν από τις στατιστικές του συνόλου δεδομένων. Το σύνολο αυτό προσφέρει ένα περιβάλλον δοκιμών μικρής κλίμακας για απλές υλοποιήσεις ορισμένων μεθόδων, ως πρώτο βήμα πριν από την κλιμάκωση σε σύνολα δεδομένων με μεγαλύτερες υπολογιστικές απαιτήσεις.

2. **Αντικειμενοκεντρικός Κωδικοποιητής Προσανατολισμένος στη Δράση.** Διεξάγουμε εκτενή πειραματισμό με μια μέθοδο απόσταξης δράσης-προς-αντικείμενο που προσπαθεί να μεταφέρει τις γνώσεις ενός προ-εκπαιδευμένου Video Masked auto-encoder σε κωδικοποιητές εικόνας. Αυτή η μέθοδος επιχειρεί να κωδικοποιήσει τις δράσεις μέσα από το Video MAE και να τις συνδέσει με την απεικόνιση των αντικειμένων που είναι στο επίκεντρο των δράσεων αυτών. Τα αποτελέσματα δείχνουν ότι οι αναπαραστάσεις του Video MAE περιέχουν χρήσιμες πληροφορίες και δοκιμάζουμε μερικές μεθόδους για να εμπλουτίσουμε τις αναπαραστάσεις των κωδικοποιητών εικόνων με αυτές. Οι μέθοδοι παρουσίασαν μια μικρή βελτίωση αλλά ίσως χρειαστούν προσαρμογές ή μεγαλύτερα σύνολα δεδομένων για την καλύτερη αξιοποίηση αυτών των αναπαραστάσεων.

3. **Slot Attention αναπαραστάσεις για κατηγοριοποίηση δυνατοτήτων [3]** . Αξιολογούμε τις αναπαραστάσεις αντικειμένων χρησιμοποιώντας ένα μοντέλο που αξιοποιεί την αρχιτεκτονική Slot Attention. Από ένα μοντέλο που έχει εκπαιδευθεί σε δεδομένα *βίντεο* εξάγουμε αντικειμενοκεντρικές αναπαραστάσεις στατικών *αντικειμένων* και προτείνουμε μια μέθοδο για τον εμπλουτισμό των διανυσμάτων αναπαράστασης των αντικειμένων με επιπλέον δρασεοκεντρική πληροφορία. Το μοντέλο παρουσιάζει ανταγωνιστική επίδοση σε σχέση με τα υπόλοιπα μοντέλα που δοκιμάστηκαν σε αυτή τη διπλωματική, ενώ επιτυγχάνει επίσης αυτόματη τμηματοποίηση των εικόνων και σημαντική μείωση στο μέγεθος αναπαράστασης ανά αντικείμενο.

Επιπλέον, η ικανότητα του μοντέλου να ανιχνεύει και να κατηγοριοποιεί πολλαπλά αντικείμενα σε μια σκηνή, παρά το γεγονός ότι έχει εκπαιδευτεί με ετικέτες και δράσεις που εστιάζουν σε ένα αντικείμενο ανά παράδειγμα, αναδεικνύει τη δυνατότητά του για γενίκευση.

4. **Slot Attention αναπαραστάσεις για ρομποτικό έλεγχο.** Παρουσιάζουμε μια μέθοδο που συνδυάζει τις χωρικές slot αναπαραστάσεων του μοντέλου SOLV [3] για τη δημιουργία αναπαραστάσεων εικόνων για χρήση σε μια προσομοίωση ρομποτικού χειρισμού. Αξιολογούμε την απόδοση αυτού του κωδικοποιητή εικόνας σε αντιπαράθεση με άλλους προ-εκπαιδευμένους κωδικοποιητές. Τα αποτελέσματα, μας δείχνουν ότι η προτεινόμενη μέθοδος σε γενικές γραμμές, επιτυγχάνει καλύτερη απόδοση σε αυτό το περιβάλλον, αυξάνοντας όμως την υπολογιστική πολυπλοκότητα.

1.3 Θεωρητικό Υπόβαθρο

Η ενότητα *Θεωρητικό Υπόβαθρο* αποσκοπεί να θέσει τις θεωρητικές βάσεις αυτής της διπλωματικής εργασίας παρέχοντας το απαραίτητο πλαίσιο που αφορά τις προτεινόμενες μεθόδους και πειράματα. Το υλικό που παρουσιάζεται αντλεί πληροφορίες από διάφορες πηγές, αλλά κυρίως από τα παρακάτω:

- Christopher M. Bishop - *Pattern Recognition and Machine Learning* [7]
- Ian Goodfellow, Yoshua Bengio, Aaron Courville - *Deep Learning* [32]
- Marco Gori's *Machine Learning: A Constraint-Based Approach* [33]
- Peter Norvig, Stuart J. Russell - *Artificial Intelligence: A Modern Approach* [93]
- Sergios Theodoridis - *Machine Learning: A Bayesian and Optimization Perspective* [107]
- Shai Shalev-Shwartz, Shai Ben-David - *Understanding Machine Learning: From Theory to Algorithms* [98]

1.3.1 Μηχανική Μάθηση

Η Μηχανική Μάθηση είναι ο επιστημονικός τομέας που επικεντρώνεται στην ανάπτυξη μεθοδολογιών που επιτρέπουν στα υπολογιστικά συστήματα να εκτελούν εργασίες μαθαίνοντας από δεδομένα, αντί να ακολουθούν ρητές οδηγίες. Η ικανότητα των αλγορίθμων μηχανικής μάθησης να μαθαίνουν από δεδομένα έχει αποδειχθεί εξαιρετικά αποτελεσματική σε τομείς όπως η αναγνώριση εικόνας και η επεξεργασία φυσικής γλώσσας, όπου οι άνθρωποι και γενικά οι βιολογικοί οργανισμοί μπορούν να εκτελούν σύνθετες εργασίες, αλλά είναι δύσκολο να εκφραστούν τα βήματα που εμπλέκονται σε αυτές σε μορφή αλγόριθμου. Επιπλέον, η μηχανική μάθηση έχει πετύχει σε εργασίες που είναι δύσκολο ή αδύνατο για στους ανθρώπους να τις εκτελέσουν, όπως η ανάλυση μεγάλων ποσοτήτων δεδομένων.

Η διάκριση ανάμεσα σε συμβολική και υπο-συμβολική Τεχνητή Νοημοσύνη [49, 33] αναδεικνύει τους τομείς που η μηχανική μάθηση έχει παρουσιάσει σημαντικά επιτεύγματα. Η συμβολική τεχνητή νοημοσύνη επικεντρώνεται στη χρήση μεθοδολογιών που εξαρτώνται από την επεξεργασία συμβόλων, προσπαθώντας να προσεγγίσει τα προβλήματα προγραμματίζοντας υπολογιστές να μιμούνται την ανθρώπινη λογική. Αυτές οι μέθοδοι έχουν το πλεονέκτημα της ερμηνευσιμότητας, καθώς το μεγαλύτερο κομμάτι των διαδικασιών είναι κατανοητή από τους ανθρώπους. Ωστόσο, επειδή τα συμβολικά συστήματα χρησιμοποιούν σύμβολα και αναπαραστάσεις υψηλού επιπέδου, συχνά απαιτούν σημαντική ανθρώπινη συμμετοχή και δυσκολεύονται σε δυναμικά περιβάλλοντα που διακατέχονται από ασάφεια και θορυβώδη δεδομένα.

Από την άλλη, ο τομέας της όρασης υπολογιστών ασχολείται με προβλήματα που είναι κυρίως υπο-συμβολικής φύσης, επειδή τα δεδομένα σε μορφή pixel δεν έχουν συμβολική σημασία και οι συμβολικοί κανόνες δεν μπορούν να εφαρμοστούν εύκολα. Η υπο-συμβολική τεχνητή νοημοσύνη, η οποία περιλαμβάνει τις περισσότερες σύγχρονες προσεγγίσεις μηχανικής μάθησης, χρησιμοποιεί

μεθόδους όπως η στατιστική εκτίμηση και η μαθηματική βελτιστοποίηση για να δημιουργήσει μοντέλα από δεδομένα. Αυτά τα μοντέλα μπορούν να εκφραστούν ως παραμετρικές συναρτήσεις και η διαδικασία εκμάθησης περιλαμβάνει τη βελτιστοποίηση των παραμέτρων χρησιμοποιώντας δεδομένα. Αυτές οι μέθοδοι δεν χαρακτηρίζονται από την ερμηνευσιμότητα των μεθόδων συμβολικής τεχνητής νοημοσύνης και αυτή η αδιαφάνεια έχει οδηγήσει στον χαρακτηρισμό τους ως *μαύρα κουτιά*. Για τον λόγο αυτό έχει αναπτυχθεί ο τομέας της *επεξηγήσιμης* μηχανικής μάθησης, ο οποίος επικεντρώνεται σε τεχνικές που παρέχουν εξηγήσεις για τις διαδικασίες και τα αποτελέσματα των μοντέλων μηχανικής μάθησης [71].

Η παραγωγή μοντέλων μηχανικής μάθησης συχνά απαιτεί μεγάλο αριθμό παραμέτρων, οι οποίες απαιτούν μεγάλο όγκο δεδομένων και υπολογιστικών πόρων για να εκπαιδευθούν. Τα τελευταία χρόνια, οι τεχνολογικές εξελίξεις στο hardware, όπως οι GPUs και οι TPUs και η διαθεσιμότητα μεγάλων συνόλων δεδομένων έχουν βοηθήσει στην υπέρβαση αυτών των εμποδίων, οδηγώντας σε σημαντικές προόδους και προωθώντας την έρευνα στον τομέα. Επίσης, τα τελευταία χρόνια, υπάρχει ενδιαφέρον για υβριδικές προσεγγίσεις που συνδυάζουν τόσο την υπο-συμβολική όσο και τη συμβολική τεχνητή νοημοσύνη, μελετώντας μεθόδους που δεν είναι μόνο αποτελεσματικές αλλά και πιο ερμηνεύσιμες.

Ταξινόμηση & Παλινδρόμηση. Τα προβλήματα στη μηχανική μάθηση κατατάσσονται σε δύο κύριες ομάδες. Εάν η επιθυμητή έξοδος είναι μια συνεχής μεταβλητή, το πρόβλημα αναφέρεται ως πρόβλημα *παλινδρόμησης*. Όταν η έξοδος είναι ένας πεπερασμένος αριθμός κατηγοριών, το πρόβλημα ονομάζεται *ταξινόμηση*. Η ταξινόμηση σε δύο κατηγορίες είναι γνωστή ως *δυαδική ταξινόμηση*. Η ταξινόμηση σε τρεις ή περισσότερες κατηγορίες αναφέρεται ως *πολυκατηγορική Ταξινόμηση*. Όταν ο στόχος είναι κάθε δείγμα να επισημαίνεται με πολλαπλές, μη αποκλειστικές ετικέτες, η εργασία ονομάζεται *Πολυετικετική (multi-label) Ταξινόμηση*. Ένα παράδειγμα multi-label ταξινόμησης είναι η *κατηγοριοποίηση προσφερόμενων δυνατοτήτων αντικειμένων*. Για αυτήν την εργασία, κάθε δείγμα είναι ένα αντικείμενο και ο στόχος είναι να προβλεφθούν οι μη αποκλειστικές προσφερόμενες δυνατότητές του (π.χ. μια μπάλα μπορεί να είναι κυλιόμενη και συμπιέσιμη).

Σύνολα Δεδομένων. Το σύνολο δεδομένων που είναι διαθέσιμο για ένα πρόβλημα συνήθως χωρίζεται σε τρία ξεχωριστά υποσύνολα: train set, validation set, test set. Καθένα από αυτά τα υποσύνολα εκτελεί έναν συγκεκριμένο ρόλο στις μεθόδους μηχανικής μάθησης. Το train set χρησιμοποιείται για τη βελτιστοποίηση των παραμέτρων του μοντέλου. Το validation set χρησιμοποιείται για επίβλεψη κατά τη διάρκεια της εκπαίδευσης για το πώς αποδίδει το μοντέλο σε άγνωστα δεδομένα. Η επίβλεψη αυτή χρησιμοποιείται για την προσαρμογή των παραμέτρων της διαδικασίας εκπαίδευσης, που αναφέρονται ως *υπερπαράμετροι*, χωρίς να επηρεάζεται αρνητικά η τελική αξιολόγηση. Το test set χρησιμοποιείται για την τελική αξιολόγηση, παρέχοντας μια εκτίμηση της απόδοσης του μοντέλου σε άγνωστα δεδομένα.

Υπερπροσαρμογή & Υποπροσαρμογή. Ο κύριος στόχος των μεθόδων μηχανικής μάθησης είναι να παράγουν μοντέλα που αποδίδουν καλά σε άγνωστα δεδομένα που προέρχονται από την ίδια κατανομή δεδομένων. Αυτή η ικανότητα είναι γνωστή και ως ικανότητα γενίκευσης. Η γενίκευση αξιολογείται με την εκπαίδευση ενός μοντέλου, βάσει μετρικών απόδοσης που υπολογίζονται στο train set, αλλά αξιολογώντας το στο test set. Επιπλέον, η ικανότητα γενίκευσης ενός μοντέλου επηρεάζεται από τη *χωρητικότητά* του (capacity), η οποία είναι η ικανότητά του να προσαρμόζεται σε πολύπλοκα σύνολα δεδομένων προσεγγίζοντας πολύπλοκες συναρτήσεις. Η μεγάλη χωρητικότητα μπορεί να φαίνεται αρχικά ως πλεονέκτημα, αλλά αυτή η ευελιξία μπορεί να οδηγήσει σε ένα φαινόμενο που ονομάζεται *υπερπροσαρμογή*, όπου το μοντέλο προσαρμόζεται υπερβολικά στο σύνολο εκπαίδευσης, μειώνοντας την απόδοσή του σε άγνωστα δεδομένα. Η υποπροσαρμογή συμβαίνει

όταν μοντέλα χαμηλής χωρητικότητας έχουν χαμηλή απόδοση, επειδή η πολυπλοκότητα της εργασίας απαιτεί μεγαλύτερη ικανότητα αναπαράστασης. Σε ορισμένα μοντέλα Μηχανικής Μάθησης, όπως τα νευρωνικά δίκτυα, ο βαθμός υπερπροσαρμογής επηρεάζεται επίσης από υπερπαραμέτρους εκπαίδευσης όπως η διάρκεια εκπαίδευσης, ο ρυθμός μάθησης κ.λ.π. [32].

Μέθοδοι Μηχανικής Μάθησης. Οι παρακάτω μέθοδοι μηχανικής μάθησης καθορίζονται από τους διαφορετικούς τύπους επίβλεψης στους οποίους έχουν πρόσβαση τα μοντέλα κατά τη διάρκεια της εκπαίδευσης:

- *Επιβλεπόμενη Μάθηση.* Σε αυτό τον τύπο μηχανικής μάθησης, οι μέθοδοι χρησιμοποιούν ανάδραση υπό τη μορφή ετικετών. Ένα επισημειωμένο σύνολο δεδομένων περιλαμβάνει μια ετικέτα για κάθε δείγμα, $\mathcal{X} = \{(x_1, y_1), \ldots, (x_N, y_N)\}$.

- *Μη Επιβλεπόμενη Μάθηση.* Στη Μη Επιβλεπόμενη Μάθηση, δεν υπάρχει άμεση ανατροφοδότηση που να καθοδηγεί τη διαδικασία εκπαίδευσης. Τα μοντέλα προσπαθούν να εντοπίσουν μοτίβα στα σύνολα δεδομένων χωρίς τη χρήση ετικετών.

- *Αυτό-επιβλεπόμενη Μάθηση.* Αυτή η προσέγγιση μηχανικής μάθησης, γενικά θεωρείται υποκατηγορία της μη επιβλεπόμενης μάθησης, και το μοντέλο παράγει τη δική του επίβλεψη από τα δεδομένα. Αυτή η τεχνική χρησιμοποιείται συχνά για την προ-εκπαίδευση μοντέλων κωδικοποιητών που στη συνέχεια βελτιστοποιούνται χρησιμοποιώντας επιβλεπόμενη μάθηση. Οι αυτο-επιβλεπόμενες μέθοδοι που χρησιμοποιούν μοντέλα κωδικοποιητών νευρωνικών δικτύων ανήκουν στο πλαίσιο ενδιαφέροντος αυτής της διπλωματικής και εξετάζονται με περισσότερη λεπτομέρεια στη συνέχεια.

- *Ενισχυτική Μάθηση.* Στην ενισχυτική μάθηση, τα μοντέλα μαθαίνουν αλληλεπιδρώντας με ένα περιβάλλον και η ανατροφοδότηση παίρνει τη μορφή μιας συνάρτησης ανταμοιβής. Αυτός ο τύπος μηχανικής μάθησης είναι εμπνευσμένος από τον τρόπο που οι άνθρωποι μαθαίνουν και αλληλεπιδρούν με τα περιβάλλοντά τους και έχει πολλές εφαρμογές στον τομέα της ρομποτικής. Η ενισχυτική μάθηση παρουσιάζεται με περισσότερη λεπτομέρεια σε επόμενο κεφάλαιο.

Μέτρα Απόδοσης και Συναρτήσεις Κόστους. Τα *μέτρα απόδοσης* είναι συναρτήσεις που ποσοτικοποιούν την απόδοση του μοντέλου και αποτελούν σημαντικό μέρος της θεωρίας και των μεθόδων Μηχανικής Μάθησης. Για εργασίες δυαδικής ταξινόμησης, όπου κάθε δείγμα μπορεί είτε να ανήκει ($y_i = 1$), είτε όχι ($y_i = 0$), σε μία μόνο κατηγορία, ορισμένες από αυτές τις συναρτήσεις βασίζονται στους ακόλουθους αριθμούς:

- *True Positives* (*TP*): Ο αριθμός των δειγμάτων για τα οποία $y_i = 1$ και $h_\vartheta(x_i) = 1$
- *True Negatives*(*TN*): Ο αριθμός των δειγμάτων για τα οποία $y_i = 0$ και $h_\vartheta(x_i) = 0$
- *False Positives* (*FP*): Ο αριθμός των δειγμάτων για τα οποία $y_i = 1$ και $h_\vartheta(x_i) = 0$
- *False Negatives* (*FN*): Ο αριθμός των δειγμάτων για τα οποία $y_i = 0$ και $h_\vartheta(x_i) = 1$

Με βάση αυτά, μπορούν να υπολογιστούν διάφορα μέτρα απόδοσης, όπως:

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN} \times 100\% \tag{1}$$

$$\textit{Recall} = \frac{TP}{TP + \mathbf{FN}} \times 100\% \tag{2}$$

$$\textit{Precision} = \frac{TP}{TP + \mathbf{FP}} \times 100\% \tag{3}$$

$$F_1 = 2 \cdot \frac{Precision \cdot Recall}{Precision + Recall} \tag{4}$$

Οι *συναρτήσεις κόστους* είναι μια υποκατηγορία των μέτρων απόδοσης που αξιολογούν πόσο απέχουν οι προβλέψεις του μοντέλου από τις πραγματικές ετικέτες και είναι συνήθως διαφορίσιμες ως προς τις παραμέτρους των μοντέλων. Σε απλά γραμμικά μοντέλα, επιτρέπουν την εκτίμηση των παραμέτρων, χρησιμοποιώντας αναλυτική διαφόριση και λύσεις κλειστής μορφής. Σε πιο πολύπλοκα μη γραμμικά μοντέλα, επιτρέπουν την εφαρμογή μεθόδων βελτιστοποίησης όπως οι επαναληπτικοί αλγόριθμοι της οικογένειας *Gradient Descent* .

1.3.2 Βαθιά Μάθηση

Η βαθιά μάθηση είναι ένας κλάδος της μηχανικής μάθησης που παρουσιάζει πολλές επιτυχίες τα τελευταία χρόνια σε τομείς όπως η όραση υπολογιστών και η επεξεργασία φυσικής γλώσσας. Ένα από τα κύρια πλεονέκτημα των τεχνικών βαθιάς μάθησης είναι η ικανότητά τους να μαθαίνουν αυτόματα αναπαραστάσεις από τα δεδομένα.

Η βαθιά μάθηση βασίζεται στους αλγόριθμους εκμάθησης *νευρωνικών δικτύων*. Τα νευρωνικά δίκτυα αποτελούνται από πολλαπλά επίπεδα. Κάθε επίπεδο χρησιμοποιεί μη γραμμικούς μετασχηματισμούς για να επεξεργαστεί την είσοδό του και παράγει μια έξοδο που μεταδίδεται στο επόμενο επίπεδο. Το πρώτο και το τελευταίο επίπεδο αναφέρονται ως επίπεδα *εισόδου* και *εξόδου* αντίστοιχα, ενώ τα ενδιάμεσα επίπεδα ονομάζονται *κρυφά* επίπεδα. Το *βάθος* του νευρωνικού δικτύου είναι ο αριθμός των επιπέδων που περιέχει και το *πλάτος* του είναι το μέγεθος των κρυφών επιπέδων του. Η αύξηση του πλάτους και του βάθους ενός νευρωνικού δικτύου γενικά ενισχύει την ικανότητά του να προσεγγίζει πιο πολύπλοκες συναρτήσεις. Ωστόσο, αυτό απαιτεί επίσης περισσότερα δεδομένα και υπολογιστικούς πόρους για τη βελτιστοποίηση των παραμέτρων.

Stochastic Gradient Descent. Ο αλγόριθμος αυτός και οι παραλλαγές του είναι οι πιο ευρέως χρησιμοποιούμενοι αλγόριθμοι βελτιστοποίησης στη βαθιά μάθηση. Τα βάρη του μοντέλου ενημερώνονται επαναληπτικά, χρησιμοποιώντας κλίσεις της συνάρτησης κόστους ως προς τις παραμέτρους. Ο υπολογισμός πραγματοποιείται σε μικρά τυχαία υποσύνολα δεδομένων που ονομάζονται mini-batches. Η κλίση, καθώς είναι ένα διάνυσμα που δείχνει προς την κατεύθυνση της ταχύτερης ανόδου της συνάρτησης απώλειας, χρησιμοποιείται για την ενημέρωση των παραμέτρων του δικτύου.

Vision Transformer. Σε αυτή τη διπλωματική, δίνεται κύρια έμφαση στα μοντέλα που βασίζονται στην αρχιτεκτονική του *Transformer* [111]. Ο κωδικοποιητής Vision Transformer (ViT) εφαρμόζει την αρχιτεκτονική transformer σε εισόδους σε μορφή εικόνας ή βίντεο. Το κύριο πλεονέκτημα της αρχιτεκτονικής αυτής είναι η ικανότητά της να επεξεργάζεται τις ακολουθίες δεδομένων παράλληλα, χωρίς να βασίζεται πάνω σε δομικές υποθέσεις για τη μορφή της εισόδου όπως άλλες επικρατούσες αρχιτεκτονικές, σαν το Convolutional Neural Network . Αυτό επιτρέπει στα transformer μοντέλα να εκπαιδεύονται σε περισσότερα δεδομένα σε λιγότερο χρόνο, να κλιμακώνονται σε πολύ μεγαλύτερα μεγέθη και να εκμεταλλεύονται καλύτερα τις μακριές εξαρτήσεις στα δεδομένα.

Με βάση την αρχιτεκτονική αυτή, η εικόνα χωρίζεται σε τμήματα (patches) σταθερού μεγέθους και κάθε τμήμα αποτελεί ένα token. Ο ViT επεξεργάζεται το σύνολο των tokens χρησιμοποιώντας την τεχνική του self-attention. Η έξοδος του ViT αποτελείται από διανύσματα αναπαράστασης, κάθε ένα αρχικοποιημένο με βάση ένα token. Το κάθε τελικό διάνυσμα αναπαραστάσεων δυνητικά εμπεριέχει πληροφορία από υπόλοιπα tokens της εικόνας. Οι *Transformers* μπορούν να εκπαιδευτούν χρησιμοποιώντας επισημειωμένα σύνολα δεδομένων σε ένα πλαίσιο επιβλεπόμενης μάθησης,

αλλά χρησιμοποιούνται επίσης σε πλαίσια αυτο-επιβλεπόμενης μάθησης αναπαραστάσεων. Τα περισσότερα μοντέλα που σχετίζονται με αυτή τη διπλωματική ακολουθούν την τελευταία προσέγγιση και, πιο συγκεκριμένα, τη μέθοδο της *αυτο-κωδικοποίησης* όπου ο *κωδικοποιητής* του *transformer* ακολουθείται από έναν *αποκωδικοποιητή* που προσπαθεί να ανακατασκευάσει την είσοδο.

1.3.3 Εκμάθηση Αναπαραστάσεων

Η εκμάθηση αναπαραστάσεων είναι ο τομέας που στοχεύει στην ανάπτυξη μεθόδων με τις οποίες τα μοντέλα εξάγουν αυτόματα χρήσιμες αναπαραστάσεις από τα δεδομένα εισόδου. Στη βαθιά μάθηση, οι μέθοδοι εκμάθησης αναπαραστάσεων συχνά χρησιμοποιούν αυτο-επιβλεπόμενη μάθηση, εκπαιδεύοντας κωδικοποιητές σε *προκαταρκτικές εργασίες* (pretext tasks) που δεν απαιτούν επισημειωμένα σύνολα δεδομένων.

Αυτο-κωδικοποιητής. Η αυτο-κωδικοποίηση είναι μία από τις επικρατέστερες μεθόδους εκμάθησης αναπαραστάσεων με χρήση αυτο-επιβλεπόμενης μάθησης. Γενικά, η μάθηση αναπαραστάσεων στοχεύει στην εκπαίδευση ενός κωδικοποιητή $e : \mathcal{D} \to \mathcal{Z}$ που απεικονίζει δεδομένα εισόδου, $x \in \mathcal{D}$, σε χρήσιμα διανύσματα αναπαράστασης, $z \in \mathcal{Z}$. Στους αυτο-κωδικοποιητές, η είσοδος ανακατασκευάζεται από μια μονάδα που ονομάζεται *αποκωδικοποιητής*, η οποία μπορεί να εκφραστεί ως συνάρτηση $g : \mathcal{Z} \to \mathcal{D}$. Μια συνηθισμένη συνάρτηση απώλειας για την εκπαίδευση αυτο-κωδικοποιητών είναι η *απώλεια ανακατασκευής*, που συνήθως ορίζεται ως η διαφορά μεταξύ της εισόδου και της ανακατασκευασμένης εξόδου (Εξίσωση 5).

$$L_r = \frac{1}{N} \sum_{i=1}^{N} \|x_i - g(e(x_i))\|, \tag{5}$$

Οι πιο ευρέως χρησιμοποιούμενοι αυτο-κωδικοποιητές είναι αυτοί που ονομάζονται *υποπλήρεις* (undercomplete) οι οποίοι επιχειρούν να ανακατασκευάσουν την είσοδο αφού την μεταφέρουν πρώτα σε έναν χώρο αναπαράστασης σημαντικά μικρότερης διάστασης. Για να ωθήσουν τα μοντέλα να εξάγουν χρήσιμες αναπαραστάσεις, έχουν προταθεί διάφορες τροποποιήσεις του υποπλήρους αυτο-κωδικοποιητή. Δύο αξιοσημείωτοι τύποι είναι οι *Αποθορυβοποιητικοί* (Denoising) και οι αυτο-κωδικοποιητές μεθόδου απόκρυψης (Masked Auto-encoders).

Οι *Αποθορυβοποιητικοί Αυτο-κωδικοποιητές* (Denoising Autoencoders) μαθαίνουν από δεδομένα που έχουν αλλοιωθεί από θόρυβο. Μια θορυβώδης εκδοχή της εισόδου, $\tilde{x}$, τροφοδοτείται στον κωδικοποιητή, και το μοντέλο καλείται να ανακατασκευάσει την αρχική είσοδο, x. Η απώλεια ανακατασκευής σε αυτή την περίπτωση είναι:

$$L_r = \frac{1}{N} \sum_{i=1}^{N} \|x_i - g(e(\tilde{x}_i))\|, \tag{6}$$

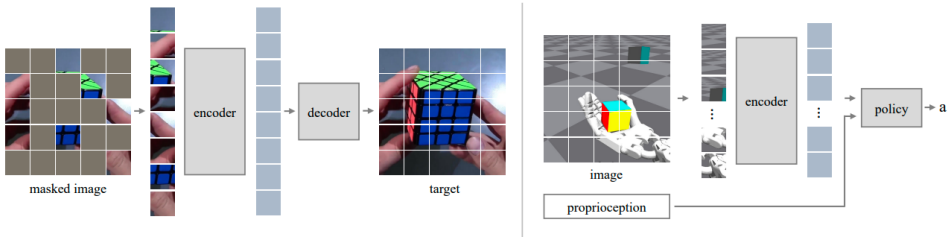
Οι *αυτο-κωδικοποιητές μεθόδου απόκρυψης* (Masked Auto-encoders) επιχειρούν να ανακατασκευάσουν μια είσοδο, $\tilde{x}$, της οποίας κάποια μέρη έχουν κρυφτεί. Στην όραση υπολογιστών, τα μοντέλα αυτά είναι ιδιαίτερα αποτελεσματικά τόσο στην κωδικοποίηση εικόνας, όσο και βίντεο και αποτελούν τα κύρια μοντέλα που χρησιμοποιούνται σε αυτή τη διπλωματική. Περισσότερες λεπτομέρειες και συγκεκριμένες τεχνικές παρουσιάζονται σε επόμενο κεφάλαιο.

Αντικειμενοκεντρικές αναπαραστάσεις. Η αντικειμενοκεντρική εκμάθηση αναπαραστάσεων είναι ένας αναπτυσσόμενος τομέας στην όραση υπολογιστών, όπου ο στόχος είναι η τμηματοποίηση οπτικών εισόδων σε αντικείμενα και η εξαγωγή αναπαραστάσεων με βάση αυτά. Οι μέθοδοι αυτές είναι συμβατές με τις *αρχές ομαδοποίησης* από την ψυχολογία, οι οποίες εξηγούν πώς οι άνθρωποι επεξεργάζονται οπτικά σήματα οργανώνοντάς τα σε αντικείμενα [77].

Η αντικειμενοκεντρική εκμάθηση αναπαραστάσεων βελτιώνει την ικανότητα γενίκευσης και την αποδοτικότητα των μοντέλων ως προς τα δείγματα εκπαίδευσης και βελτιώνει την ερμηνευσιμότητα. Στο παρελθόν όμως, αυτές οι τεχνικές βασίζονταν στο μοντέλο της επιβλεπόμενης μάθησης και περιορίζονταν από τη δυσκολία και το κόστος της επισήμανσης συνόλων δεδομένων. Τα τελευταία χρόνια, αρχιτεκτονικές όπως το Slot Attention, οι οποίες είναι αυτο-επιβλεπόμενες και εξαιρετικά κλιμακώσιμες, έχουν επιταχύνει τις εξελίξεις σε αυτόν τον χώρο εκμεταλλευόμενες μεγάλα μη επισημειωμένα σύνολα δεδομένων εικόνων και βίντεο [67, 8, 3].

Εκμάθηση αναπαραστάσεων στη ρομποτική Στον τομέα της ρομποτικής, τα τελευταία χρόνια έχει ενταθεί το ενδιαφέρον σε προσεγγίσεις βασισμένες σε μεθόδους μηχανικής και βαθιάς μάθησης. Η μεταφορά γνώσεων από την επιτυχημένη εφαρμογή της εκμάθησης αναπαραστάσεων στην επεξεργασία φυσικής γλώσσας και την όραση υπολογιστών παίζει κεντρικό ρόλο σε αυτές τις προσπάθειες.

Οι αναπαραστάσεις εικόνας παίζουν καθοριστικό ρόλο στα προβλήματα ρομποτικών χειρισμών, όπου η κατανόηση του περιβάλλοντος του ρομπότ είναι καθοριστική. Ειδικότερα, με την πρόοδο της βαθιάς μάθησης, η όραση υπολογιστών ενισχύθηκε με αποτελεσματικές τεχνικές εξαγωγής αναπαραστάσεων. Οι τεχνικές αυτές είναι η βάση για κάποιες από τις πιο επιτυχημένες μεθόδους εκμάθησης οπτικοκινητικών πολιτικών (Σχήμα 2) [64, 106, 61, 95, 128]. Ο όρος *οπτικοκινητικός* επισημαίνει ότι το διάνυσμα κατάστασης που δίνεται ως είσοδος στο μοντέλο πολιτικής συνδυάζει αναπαραστάσεις εικόνας με ένα διάνυσμα που περιλαμβάνει πληροφορίες για την κατάσταση του ρομπότ, όπως η θέση ή οι ταχύτητες των αρθρώσεών του.



Σχήμα 2: Εκμάθηση αναπαραστάσεων με αυτο-κωδικοποιητή με μέθοδο απόκρυψης εικόνας για τον έλεγχο ρομπότ. Πηγή: [87].

Οι τεχνικές αυτο-επιβλεπόμενης μάθησης είναι πολύ σημαντικές στην εκμάθηση αναπαραστάσεων για ρομπότ, καθώς επιτρέπουν την αξιοποίηση μεγάλων μη επισημειωμένων συνόλων δεδομένων. Ένα ενδεικτικό παράδειγμα είναι η επιτυχία των αυτο-κωδικοποιητών εικόνας σε προσομοιωμένες [87] και πραγματικού κόσμου [88] ρομποτικές εργασίες. Στη φάση προ-εκπαίδευσης, οι κωδικοποιητές εκπαιδεύονται χρησιμοποιώντας εικόνες από προσωποκεντρικά (egocentric) σύνολα δεδομένων, όπως το Ego4D[36] και το EPIC-Kitchens[17], καθώς και σύνολα δεδομένων επικεντρωμένα σε δράσεις, όπως το Something-Something [35]. Στη συνέχεια, οι παράμετροι των κωδικοποιητών μένουν σταθερές και οι αναπαραστάσεις τους χρησιμοποιούνται για την εκμάθηση οπτικοκινητικών πολιτικών ελέγχου.

1.4 Απόσταξη Γνώσης από Δράση σε Αντικείμενο

Σε αυτό το μέρος, προτείνουμε και πειραματιζόμαστε με μια διαδικασία απόσταξης γνώσης (knowledge distillation) δράση-σε-αντικείμενο που μεταφέρει τη γνώση από έναν κωδικοποιητή Video MAE σε έναν κωδικοποιητή εικόνας. Το αποτέλεσμα της διαδικασίας αυτής είναι ένας Αντικειμενοκεντρικός Κωδικοποιητής Προσανατολισμένος στη Δράση (Object Action-centric Encoder -OAcE). Ο

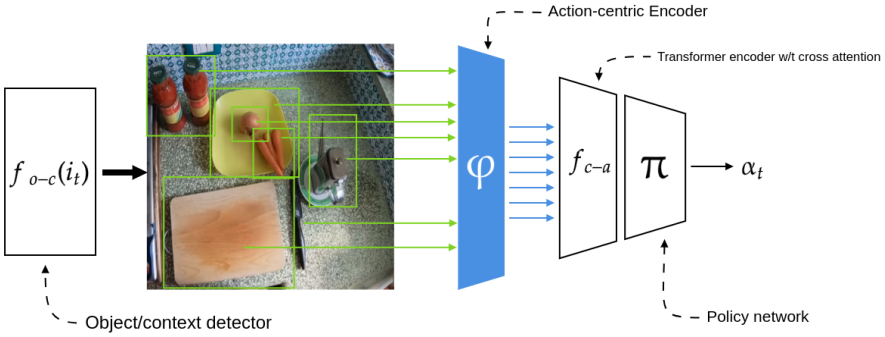
στόχος του OAcE είναι να μοντελοποιήσει τον χώρο αναπαραστάσεων βίντεο που περιέχουν δράσεις πάνω σε αντικείμενα, που είναι αρχικά προσβάσιμος μόνο από το μοντέλο Video MAE. Χρησιμοποιώντας διατροπική απόσταξη γνώσης (cross-modal distillation), στοχεύουμε να κάνουμε αυτές τις αναπαραστάσεις προσβάσιμες μέσω στατικών εικόνων αντικειμένων.

Ο OAcE αποτελείται από δύο κύρια μέρη:

- Έναν **Κωδικοποιητή Εικόνας** που μετασχηματίζει μια εικόνα από τον χώρο των pixel σε έναν πυκνό χώρο αναπαράστασης εικόνας, $I \in \mathcal{Z}img$. Στα παρακάτω πειράματα, χρησιμοποιούνται δύο διαφορετικά προ-εκπαιδευμένα μοντέλα ως Κωδικοποιητές Εικόνας, ο CLIP[86] και ο Image MAE[45].

- Ένα **Μοντέλο Αντιστοίχισης**, το οποίο αντιστοιχίζει τον χώρο αναπαράστασης εικόνας στον δρασεοκεντρικό χώρο αναπαράστασης βίντεο, $R \in \mathcal{Z}ac$.

Διαισθητικά, αυτή η μέθοδος είναι προσπάθεια κωδικοποίησης των εμπειριών δράσης και συσχέτισής τους, μέσω της χρήσης απόσταξης γνώσης, με την απεικόνιση των αντικειμένων που είναι το επίκεντρο των δράσεων αυτών. Μια μελλοντική κατεύθυνση θα μπορούσε να περιλαμβάνει τη συσχέτιση των εμπειριών των ίδιων των πρακτόρων με τα αντικείμενα.



Σχήμα 3: Παράδειγμα του OAcE κωδικοποιητή ως μέρος μοντέλου εκμάθησης πολιτικών.

Οι OAcE αναπαραστάσεις θα ήταν δυνητικά χρήσιμες ως βήμα προ-εκπαίδευσης σε μία αντικειμενοκεντρική μέθοδο εκμάθησης πολιτικών, όπως στο Σχήμα 3. Μια άλλη πιθανή χρησιμότητα της ενσωμάτωσης δρασεοκεντρικής πληροφορίας θα μπορούσε να είναι η παροχή μέτρων ομοιότητας σε ένα περιβάλλον ανάκτησης παραδειγμάτων, όπως στη διαδικτυακή βάση δεδομένων ρομποτικών χειρισμών που προτάθηκε στο [117]. Τέλος, κάποια παρόμοια ιδέα θα μπορούσε να εφαρμοστεί σε εφαρμογές Επαυξημένης ή Εικονικής Πραγματικότητας, όπου οι εικονικοί βοηθοί θα μπορούσαν να παρέχουν υποστήριξη, και ίσως να χρειαστεί να ανακτήσουν και να παρέχουν παραδείγματα δράσεων [83].

Ο κύριος στόχος αυτής της πειραματικής ενότητας είναι να διερευνήσει αν ο OAcE μπορεί να συγκριθεί και δυνητικά να βελτιώσει, ορισμένους από τους σύγχρονους κωδικοποιητές εικόνας. Πριν από την παρουσίαση της πειραματικής μεθοδολογίας και των αποτελεσμάτων, η επόμενη ενότητα παρουσιάζει το θεωρητικό υπόβαθρο που ενέπνευσε αυτή τη μελέτη, μαζί με τα προ-εκπαιδευμένα μοντέλα που χρησιμοποιούνται.

1.4.1 Απόσταξη Γνώσης

Η απόσταξη γνώσης (Knowledge Distillation - KD) [34, 53, 90] είναι μια μέθοδος συμπίεσης νευρωνικών δικτύων, στην οποία ένα μοντέλο μαθητής (student model) εκπαιδεύεται να αναπαραγάγει τη λειτουργία ενός μεγαλύτερου και πιο σύνθετου μοντέλου δασκάλου (teacher model). Αυτή η μέθοδος προτάθηκε, μαζί με άλλες τεχνικές μείωσης μοντέλων όπως το Network Pruning [14, 91, 31], για να καλύψει την ανάγκη για μοντέλα που είναι εξίσου αποτελεσματικά με τα μεγάλα βαθιά μοντέλα, αλλά λειτουργούν σε συσκευές με περιορισμένους υπολογιστικούς πόρους, όπως κινητά τηλέφωνα ή αυτοκίνητα. Το μοντέλο μαθητής δεν μαθαίνει μόνο από το σύνολο δεδομένων, αλλά και αποτυπώνει την ικανότητα γενίκευσης του δασκάλου [48].

Τα τελευταία χρόνια, έχουν προταθεί αρκετές παραλλαγές της απόσταξης γνώσης [34, 53]. Οι κατηγορίες που σχετίζονται με τα πειράματα αυτής της διπλωματικής είναι οι εξής:

- **Feature-Based Απόσταξη**: Σε αυτή την κατηγορία αλγορίθμων KD, η μεταφερόμενη γνώση είναι σε υψηλότερο επίπεδο σε σύγκριση με τις Response-Based knowledge methods, όπου το μοντέλο μαθητής στοχεύει τις πιθανότητες ταξινόμησης του μοντέλου δασκάλου. Η Feature-Based μέθοδος έχει δείξει ενθαρρυντικά αποτελέσματα ως μέθοδος εκμάθησης αναπαράστασης [112, 26, 28].

- **Διατροπική Απόσταξη (cross-modal distillation)**: Αυτό σημαίνει ότι η είσοδος του δασκάλου είναι διαφορετικής μορφής από αυτή του μαθητή. Στην προσέγγισή μας, ο δάσκαλος κωδικοποιεί βίντεο και ο μαθητής προσπαθεί να αποστάξει τις πληροφορίες που σχετίζονται με τη δράση σε εικόνες των αντικειμένων. Αυτό εμπίπτει στην κατηγορία της απόσταξης βίντεο σε εικόνα ή μεταφοράς γνώσης [92, 80, 65].

- **Σχεσιακή (Relational) Απόσταξη**: Αυτή η παραλλαγή εστιάζει στη μεταφορά των σχέσεων μεταξύ δειγμάτων στον χώρο αναπαράστασης του μοντέλου δασκάλου. Η σχέση των δειγμάτων ποσοτικοποιείται συνήθως μέσω δύο τύπων συνάρτησης απώλειας [79]: απώλεια βάσει απόστασης (distance-wise loss) και απώλεια βάσει γωνίας (angle-wise loss). Στην απώλεια βάσει απόστασης, οι Ευκλείδειες αποστάσεις μεταξύ ζευγών δειγμάτων υπολογίζονται, ενθαρρύνοντας τον μαθητή να διατηρεί σχέσεις αποστάσεων παρόμοιες με αυτές του δασκάλου. Η απώλεια βάσει γωνίας επιχειρεί μια πιο λεπτομερή μεταφορά σχεσιακής πληροφορίας, ωθώντας τον μαθητή να διατηρήσει τις γωνίες που σχηματίζονται από τριάδες παραδειγμάτων.

1.4.2 Κωδικοποιητές εικόνας

CLIP. Το μοντέλα CLIP [86] είναι από τους πιο επιτυχημένους κωδικοποιητές εικόνας , όσον αφορά τη γενίκευση, την ευελιξία και την αποδοτικότητα. Τα μοντέλα αυτά εκπαιδεύονται σε ένα σύνολο δεδομένων που αποτελείται από επισημειωμένες εικόνες. Η εκπαίδευση βασίζεται στη μέθοδο αντιθετικής μάθησης (contrastive learning framework). Το μοντέλα CLIP αποτελούνται από ένα ζευγάρι (Transformer encoders), όπου ο ένας κωδικοποιεί εικόνες και ο άλλος κείμενο. Οι αναπαραστάσεις καταλήγουν σε έναν κοινό χώρο στον οποίο η εκπαίδευσή στοχεύει να φέρει τα σωστά ζευγάρια πιο κοντά, ενώ ταυτόχρονα να απομακρύνει τα λάθος ζευγάρια, τα οποία είναι τυχαίοι συνδυασμοί κειμένου-εικόνας. Ένα προ-εκπαιδευμένο μοντέλο CLIP επιλέχθηκε ως ένας από τους κωδικοποιητές εικόνας στο OAcE, λόγω της αποδεδειγμένης του αποτελεσματικότητας.

Image Masked Auto-encoder. Η κωδικοποιητής εικόνας Image Masked Auto-encoder (MAE)[45] βασίζεται στην αυτο-κωδικοποίηση με τη μέθοδο απόκρυψης (masked auto-encoding). Η βασική ιδέα πίσω από τη μέθοδο αυτή είναι ότι αν ένα μοντέλο μπορεί να ανακατασκευάσει ένα δείγμα με ορισμένα από τα μέρη του κρυμμένα, τότε ο κωδικοποιητής του μπορεί να εξαγάγει υψηλής

ποιότητας αναπαραστάσεις. Η μέθοδος εκπαίδευσης του Image MAE βασίζεται στην αρχιτεκτονική Κωδικοποιητή-Αποκωδικοποιητή, όπου και τα δύο μοντέλα έχουν αρχιτεκτονική ViT. Ο κωδικοποιητής επεξεργάζεται την ελλιπή είσοδο και παράγει αναπαραστάσεις που έχουν ενσωματώσει πληροφορία για ολόκληρη την εικόνα. Ο αποκωδικοποιητής προσπαθεί να ανακατασκευάσει την αρχική εικόνα και η συνάρτηση απώλειας είναι το μέσο τετράγωνο σφάλμα μεταξύ των ανακατασκευασμένων και των αρχικών τιμών των pixel στα token που έχουν αποκρυφθεί.

1.4.3 Κωδικοποιητής βίντεο

Video Masked Auto-encoder. Το μοντέλο Video MAE που χρησιμοποιείται σε αυτά τα πειράματα παρουσιάστηκε στο [109]. Η μέθοδος που προτείνεται αντιμετωπίζει τις προκλήσεις που παρουσιάζουν τα δεδομένα σε μορφή βίντεο, σε σύγκριση με τις μορφές κειμένου και εικόνας. Η πρώτη πρόκληση είναι η αυξημένη πολυπλοκότητα που εισάγεται από τη διάσταση του χρόνου. Η δεύτερη πρόκληση είναι ότι, στις περισσότερες περιπτώσεις, το χρήσιμο σήμα είναι μόνο ένα μικρό ποσοστό της συνολικής εισόδου. Τέλος, όταν αποκρύπτονται κομμάτια του βίντεο, η υψηλή χρονική συσχέτιση μεταξύ των frame μπορεί να οδηγήσει σε διαρροή πληροφορίας σε μέρη του βίντεο με περιορισμένη κίνηση. Για να αντιμετωπιστούν αυτές οι προκλήσεις, οι συγγραφείς προτείνουν τη χρήση μιας τεχνικής σωληνοειδούς απόκρυψης (tube masking), όπου οι χρονικοί γείτονες ενός token είναι επίσης κρυμμένοι. Επιπλέον, για να αναγκάσουν το μοντέλο να εστιάσει στο χρήσιμο μέρος του σήματος και να αποφύγουν τις ψευδείς συσχετίσεις, αποκρύπτεται το 90-95% των συνολικών token .

Τα μοντέλα που εκπαιδεύτηκαν με αυτή τη μέθοδο παράγουν state-of-the-art αποτελέσματα στην downstream εργασία *αναγνώρισης δράσης*. Ένα από τα σύνολα δεδομένων που χρησιμοποιούνται για την αξιολόγηση των Video MAEs είναι το σύνολο δεδομένων Something-Something v.2. Αυτό το σύνολο δεδομένων επιλέχθηκε ως βάση για το κύριο μέρος των πειραμάτων αυτού του μέρους, καθώς είναι ελαφρύ και παρέχει βίντεο επικεντρωμένα σε δράσεις. Ένα Video MAE προεκπαιδευμένο σε αυτό το σύνολο δεδομένων χρησιμοποιείται ως μοντέλο δασκάλου στη διαδικασία απόσταξης.

1.4.4 Προσφερόμενες Δυνατότητες Αντικειμένων

Στο πλαίσιο των οπτικών αναπαραστάσεων, η έννοια των προσφερόμενων δυνατοτήτων αντικειμένων (affordances), η οποία συνδέει την αντίληψη των αντικειμένων με τις δυνατότητες δράσης, παρέχει μια πολύτιμη οπτική για τις μεθόδους εκμάθησης αναπαραστάσεων, καθώς καταλαμβάνει τον χώρο μεταξύ αυτού που είναι αντικειμενικά παρατηρήσιμο (χαρακτηριστικά αντικειμένων) και αυτού που βιώνεται υποκειμενικά (αναπαραστάσεις) [76, 13]. Ο James J. Gibson υποστήριξε ότι για τους ανθρώπους και τα ζώα, τα αντικείμενα δεν γίνονται απλώς αντιληπτά ως συνθέσεις των χαρακτηριστικών τους (σχήμα, χρώμα, υφή), αλλά ως συνθέσεις των δυνατοτήτων δράσης που παρέχουν [30, 76].

Τα βασικά προβλήματα που σχετίζονται με αυτή τη διπλωματική ορίζονται παρακάτω.

- Κατηγοριοποίηση προσφερόμενων δυνατοτήτων: Αυτή περιλαμβάνει τη multi-label ταξινόμηση των εικόνων σε ένα σύνολο διαθέσιμων προσφερόμενων δυνατοτήτων. Αυτή η εργασία είναι συνήθως βάση για πιο σύνθετες εργασίες αναγνώρισης προσφερόμενων δυνατοτήτων.

- Ανίχνευση προσφερόμενων δυνατοτήτων: Στην εργασία αυτή τα μοντέλα πρέπει να *εντοπίσουν* και να κατηγοριοποιήσουν τα αντικείμενα με βάση τις προσφερόμενες δυνατότητές τους.

Η κατηγοριοποίηση προσφερόμενων δυνατοτήτων είναι ένας καλός υποψήφιος για την αξιολόγηση αναπαραστάσεων αντικειμένων που προορίζονται για ρομποτική. Αυτό οφείλεται στο γεγονός ότι

ο εντοπισμός δυνητικών δράσεων σε ένα περιβάλλον μπορεί να βοηθήσει το ρομποτ να σχεδιάσει και να συνεργαστεί με ανθρώπους ή άλλα ρομπότ [13, 42].

1.4.5 Σύνολο Δεδομένων

Τα Something's Affordances είναι ένα μικρής κλίμακας σύνολο δεδομένων που βασίζεται στο σύνολο δεδομένων Something-Something v.2 [35] χρησιμοποιώντας κάποιες από τις bounding box επισημειώσεις του συνόλου Something-Else [72]. Οι κατηγορίες δράσης στο Something-Something v.2 έχουν δημιουργηθεί με στόχο να βοηθήσουν τα μοντέλα, να εμβαθύνουν την κατανόησή τους για τον φυσικό κόσμο και να αναπτύξουν μια μορφή κοινής λογικής. Οι αναπαραστάσεις του VideoMAE αποδίδουν καλά σε αυτό το σύνολο δεδομένων και συνεπώς, είναι πιθανό να έχουν αποτυπώσει μια υψηλής ποιότητας δρασεοκεντρική πληροφορία.

Το Something's Affordances εστιάζει στο πρόβλημα της κατηγοριοποίηση προσφερόμενων δυνατοτήτων. Για την αξιολόγηση των μεθόδων υπό εξέταση επιλέχθηκε ένα μικρό υποσύνολο κατηγοριών δράσης. Οι κατηγορίες αυτές και οι αντίστοιχες προσφερόμενες δυνατότητες παρουσιάζονται στον παρακάτω πίνακα.

Affordance	**Something-Something action labels**	**# video samples**
Foldable	Folding something, Unfolding something	1620
Rollable	Rolling something on a flat surface, Letting something roll up a slanted surface, so it rolls back down, Letting something roll down a slanted surface, Letting something roll along a flat surface	2913
Squeezable	Squeezing something	2202
Containment	Pouring something out of something, Pouring something into something until it overflows, Pretending to pour something out of something, but something is empty, Showing that something is empty	2289
Tearable	Tearing something just a little bit	1620

Table 1: Οι κατηγορίες δράσης του Something's Affordances και οι αντίστοιχες προσφερόμενες δυνατότητες.

Ένας από τους περιορισμούς στην εξαγωγή bounding box αντικειμένων από ένα σύνολο δεδομένων βίντεο είναι ότι πολλά δείγματα περιέχουν παρεμβολές από χέρια ή άλλα αντικείμενα. Για να ελαχιστοποιηθεί αυτό το ζήτημα, οι εικόνες αντικειμένων εξήχθησαν από τα πρώτα 10 frame των βίντεο, όπου τα αντικείμενα συνήθως εμφανίζονται μόνα τους. Επιπλέον, λόγω της κίνησης της κάμερας ή των χεριών, κάποια από τα bounding box περιέχουν μέρος του αντικειμένου ή εμφανίζουν motion blur. Αυτό έρχεται σε αντίθεση με άλλα σύνολα δεδομένων κατηγοριοποίησης προσφερόμενων δυνατοτήτων, όπως το [55], που περιέχουν καθαρές εικόνες αντικειμένων. Αν και αυτό μπορεί αρχικά να φαίνεται ως ένα μειονέκτημα, αυτές οι αλλοιώσεις μπορούν να προσομοιώσουν τα αποτελέσματα τεχνικών αύξησης εικόνας, οι οποίες χρησιμοποιούνται τεχνητά για να βελτιώσουν την ικανότητα γενίκευσης των μοντέλων [116].

Παρομοίως με το σύνολο δεδομένων Something-Else [72], ορίζουμε το υποσύνολο *frequent objects* , το οποίο αποτελείται από τα αντικείμενα που εμφανίζονται περισσότερες από 20 φορές στα βίντεο. Αυτό γίνεται για να διασφαλιστεί ότι τα αντικείμενα εμφανίζονται σε αρκετά παραδείγματα, ώστε να μπορεί να εξαχθεί πληροφορία προσφερόμενων δυνατοτήτων από τα στατιστικά του συνόλου δεδομένων. Συνολικά, το σύνολο δεδομένων αποτελείται από $11,235$ βίντεο, από τα οποία εξάγονται $123,434$ bounding boxes αντικειμένων. Για κάθε αντικείμενο στο σύνολο *frequent objects* υπολογίζουμε την κατανομή συχνότητας των δράσεων. Από αυτήν την κατανομή συχνότητας, εξάγουμε τις multi-label προσφερόμενες δυνατότητες για κάθε αντικείμενο, εφαρμόζοντας ένα κατώφλι στις

συχνότητες με τρόπο ώστε να αποφεύγονται τα αντικείμενα που χρησιμοποιούνται με ασυνήθιστο τρόπο. Για παράδειγμα, η κατανομή συχνότητας δράσης και οι multi-label προσφερόμενες δυνατότητες για το αντικείμενο *bottle* παρουσιάζονται στον παρακάτω πίνακα:

	foldable	rollable	squeezable	containment	tearable
frequency distribution	2	575	156	178	1
affordance	0	1	1	1	0

Table 2: Η κατανομή συχνότητας δράσης και οι multi-label προσφερόμενες δυνατότητες για το αντικείμενο *bottle*.

Το σύνολο δεδομένων χωρίζεται με δύο τρόπους:

1. **Διαίρεση βάσει βίντεο (SA-vb)**: Το σύνολο δεδομένων χωρίζεται σε τρία σύνολα train, validation, test εξασφαλίζοντας ότι οι εικόνες από το ίδιο βίντεο ανήκουν στο ίδιο σύνολο.
2. **Διαίρεση βάσει αντικειμένων (SA-ob)**: Αυτή η διαίρεση στοχεύει στη συνθετική γενίκευση compositional generalization [72] , διαιρώντας τα αντικείμενα σε δύο σύνολα, το *Set A* και το *Set B*. Το *Set A* χρησιμοποιείται για την εκπαίδευση, ενώ το *Set B* χρησιμοποιείται στα σύνολα validation και test .

Τα ακόλουθα πειράματα αποτελούνται από δύο στάδια. Στο αρχικό στάδιο, ο κωδικοποιητής OAcE εκπαιδεύεται χρησιμοποιώντας εικόνες αντικειμένων ως εισόδους και αναπαραστάσεις βίντεο από τον Video MAE ως στόχους. Για να επιταχυνθεί αυτή η διαδικασία, οι αναπαραστάσεις τόσο του Κωδικοποιητή Εικόνας όσο και του Κωδικοποιητή Βίντεο, εξάγονται εκ των προτέρων, καθώς μόνο το Μοντέλο Αντιστοίχισης υποβάλλεται σε εκπαίδευση. Στο δεύτερο στάδιο, ο εκπαιδευμένος κωδικοποιητής δοκιμάζεται στην κατηγοριοποίηση προσφερόμενων δυνατοτήτων χρησιμοποιώντας τους multi-label στόχους που περιγράφηκαν παραπάνω.

1.4.6 Πειραματική μέθοδος

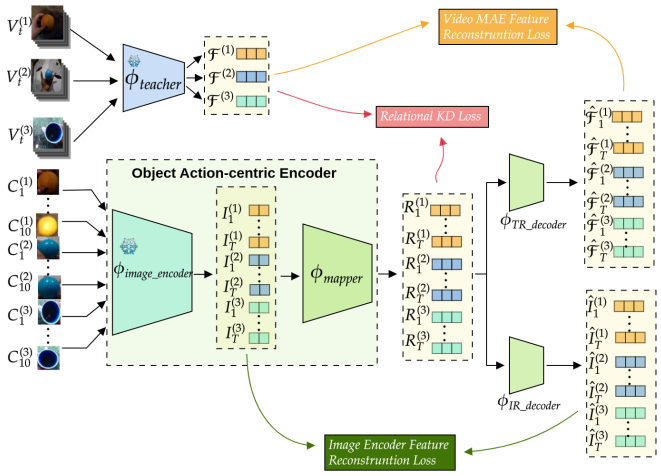
Η αρχιτεκτονική και η μέθοδος εκπαίδευσης του OAcE απεικονίζεται στο Σχήμα 4. Ο OAcE λαμβάνει ως είσοδο, εικόνες αντικειμένων, τα οποία εξάγονται χρησιμοποιώντας τα bounding boxes από το σύνολο δεδομένων Something-Else. Τα βίντεο των δράσεων ανήκουν στο Something's Affordance. Για κάθε εικόνα αντικειμένου, ο OAcE εκπαιδεύεται για να παράγει μία αναπαράσταση στον δρασεοκεντρικό χώρο αναπαράστασης .

Κωδικοποίηση από τον δάσκαλο. Το μοντέλο δάσκαλος είναι το προ-εκπαιδευμένο ViT-S Video MAE από το [109]. Θεωρούμε τα βίντεο στο σύνολο δεδομένων Something's Affordances ως $\mathcal{X}_t^{(i)}, i \in [1..N]$, καθένα από τα οποία αποτελείται από $T^{(i)}$ frames (Εξίσωση 7). Τα frames έχουν σταθερό ύψος 224 pixels και μεταβλητό πλάτος. Πριν από την εισαγωγή στο Video MAE, μετασχηματίζονται σε σταθερή ανάλυση 224×224 ($H = W = 224$). Επιπλέον, τα βίντεο υποβάλλονται σε χρονική υποδειγματολειψία, καταλήγοντας σε video clip με 16 frame (Εξίσωση 8). Το Video MAE επεξεργάζεται τα video clip και οι αναπαραστάσεις $\mathcal{F}^{(i)}$ προέρχονται από το average pooling των tokens του ViT . Το μέγεθος αυτών των διανυσμάτων αναπαράστασης είναι $d_t = 384$.

$$\textbf{Videos: } \mathcal{X}_t^{(i)} = \{\mathbf{x}_1, \ldots, \mathbf{x}_{T^{(i)}}\} \in \mathbb{R}^{T^{(i)} \times H \times W \times 3} \tag{7}$$

$$\textbf{Video Clips: } V_t^{(i)} = \{\mathbf{v}_1, \ldots, \mathbf{v}_{16}\} \in \mathbb{R}^{16 \times H \times W \times 3} \tag{8}$$

$$\textbf{Teacher representations: } \mathcal{F}(i) = \phi_{teacher}(V_t^{(i)}) \in \mathbb{R}^{d_t} \tag{9}$$



Σχήμα 4: Object Action-Centric Encoder: αρχιτεκτονική και μέθοδος εκπαίδευσης

Κωδικοποίηση εικόνας. Δύο προεκπαιδευμένοι κωδικοποιητές εικόνας χρησιμοποιήθηκαν σε αυτήν την πειραματική ενότητα, ένας CLIP [86] και ένας Image MAE [45]. Ο Image MAE εκπαιδεύτηκε επιπλέον στις εικόνες του συνόλου δεδομένων Something's Affordances για 100 εποχές.

Όπως αναφέρθηκε προηγουμένως, για να μειώσουμε την οπτική παρεμβολή, τα αποκομμένα αντικείμενα, $Ct^{(i)}$ (Εξίσωση 10), εξάγονται από τα πρώτα 10 frame των βίντεο, χρησιμοποιώντας τα boudning boxes από το σύνολο δεδομένων Something-Else. Τα αποκομμένα αντικείμενα υποβάλλονται στη συνέχεια σε επεξεργασία από τον προ-επεξεργαστή κάθε κωδικοποιητή εικόνας. Και οι δύο κωδικοποιητές δέχονται εικόνες σχήματος $H \times W \times 3$, όπου $H = W = 224$. Ο κωδικοποιητής εικόνας επεξεργάζεται τα αποκομμένα αντικείμενα για να παράγει τις αναπαραστάσεις εικόνας $It^{(i)}$. Και στις δύο περιπτώσεις, το μέγεθος των διανυσμάτων αναπαράστασης εικόνας είναι $d_i = 512$.

$$\textbf{Object crops: } C_t^{(i)} = \left\{\mathbf{C}_1^{(i)}, \ldots, \mathbf{C}_T^{(i)}\right\} \in \mathbb{R}^{10\times H\times W\times 3} \tag{10}$$

$$\textbf{Image representations: } I_t^{(i)} = \phi_{image\_encoder}(C_t^{(i)}) \in \mathbb{R}^{d_i} \tag{11}$$

Αντιστοίχιση στον δρασεοκεντρικό χώρο αναπαράστασης. Η αντιστοίχιση από τις αναπαραστάσεις εικόνας, $I_t^{(i)}$, στις αναπαραστάσεις OAcE, $R_t^{(i)}$, παράγεται από ένα MLP (Εξίσωση 12) που αποτελείται από τα ακόλουθα στρώματα, συνδεδεμένα σε σειρά:

1. Ένα γραμμικό επίπεδο με μέγεθος εισόδου 512 και μέγεθος εξόδου 512
2. Ένα επίπεδο ενεργοποίησης ReLU
3. Ένα επίπεδο (dropout)
4. Ένα γραμμικό επίπεδο με μέγεθος εισόδου 512 και μέγεθος εξόδου 384

$$\textbf{OAcE representations: } R_t^{(i)} = \phi_{mapper}(\mathcal{I}_t^{(i)}) \in \mathbb{R}^{d_i} \tag{12}$$

Αποκωδικοποίηση σε επίπεδο αναπαραστάσεων. Για την εκπαίδευση του OAcE, οι αναπαραστάσεις αποκωδικοποιούνται για να ανακατασκευάσουν τις αναπαραστάσεις Video MAE (Εξίσωση 13) και τις αναπαραστάσεις Εικόνας (Εξίσωση 14). Η ανακατασκευή και των δύο αναπαραστάσεων οδήγησε σε λίγο καλύτερα αποτελέσματα από το να έχει ο OAcE ως στόχο μόνο τα χαρακτηριστικά του Video MAE.

$$\textbf{Teacher representation (TR) reconstructions: } \hat{\mathcal{F}}_t^{(i)} = \phi_{TR\_decoder}(R_t^{(i)}) \in \mathbb{R}^{d_t} \tag{13}$$

$$\textbf{Image representation (IR) reconstructions: } \hat{I}_t^{(i)} = \phi_{IR\_decoder}(R_t^{(i)}) \in \mathbb{R}^{d_i} \tag{14}$$

Συναρτήσεις Απώλειας. Το Μοντέλο Αντιστοίχισης και οι δύο αποκωδικοποιητές βελτιστοποιούνται χρησιμοποιώντας τρεις διαφορετικές συναρτήσεις απώλειας:

1. **Απώλεια ανακατασκευής αναπαραστάσεων δασκάλου**: Αυτή είναι η Απώλεια Μέσου Τετραγωνικού Σφάλματος (*MSE*) που υπολογίζεται μεταξύ των στόχων αναπαραστάσεων από το Video MAE για κάθε βίντεο και των ανακατασκευασμένων αναπαραστάσεων των αντίστοιχων αντικειμένων στο ίδιο βίντεο. Για ένα batch B με N δείγματα βίντεο:

$$L_{TR} = \frac{1}{N \cdot d_t} \sum_{i=1}^{N} \sum_{t=1}^{10} L_{MSE}(\mathcal{F}^{(i)}, \hat{\mathcal{F}}_t^{(i)}) \tag{15}$$

2. **Απώλεια ανακατασκευής αναπαραστάσεων εικόνας**: Αυτή είναι η *MSE* που υπολογίζεται μεταξύ των αναπαραστάσεων εικόνας από τον Κωδικοποιητή Εικόνας και των ανακατασκευασμένων αναπαραστάσεων εικόνας. Για ένα batch B με N δείγματα αντικειμένων:

$$L_{IR} = \frac{1}{N \cdot d_t} \sum_{i=1}^{N} \sum_{t=1}^{10} L_{MSE}(I_t^{(i)}, \hat{I}_t^{(i)}) \tag{16}$$

3. **Απώλεια σχεσιακής απόστασης**: Αυτή είναι η Angle-wise Relational Knowledge Distillation Loss (RKD-A) όπως προτάθηκε στο [79]. Για μία τριάδα δειγμάτων, η σχεσιακή δυναμική γωνίας ποσοτικοποιεί τη γωνία που δημιουργείται από τα τρία δείγματα σε ένα χώρο αναπαραστάσεων:

$$\begin{aligned} \psi_A(R_i, R_j, R_k) &= \cos \angle R_i R_j R_k = (\mathbf{e}_{ij}, \mathbf{e}_{kj}) \\ where \quad \mathbf{e}_{ij} &= \frac{t_i - t_j}{\|t_i - t_j\|_2}, \quad \mathbf{e}_{kj} = \frac{t_k - t_j}{\|t_k - t_j\|_2}. \end{aligned} \tag{17}$$

   Η απώλεια RKD-A μετρά τη διαφορά στη σχεσιακή δυναμική γωνίας μεταξύ των αναπαραστάσεων του OAcE και των αναπαραστάσεων του δασκάλου:

$$\text{RKΔ-A} = \frac{1}{|C^3|} \sum_{(C_i, C_j, C_k) \in C^3} L_{MSE}(\psi_A(R_i, R_j, R_k), \psi_A(\mathcal{F}_i, \mathcal{F}_j, \mathcal{F}_k)) \tag{18}$$

   Για να περιοριστεί η αύξηση της υπολογιστικής πολυπλοκότητας που εισάγει αυτή η απώλεια, το C^3 είναι ένα σύνολο 50 τριάδων, που επιλέγονται τυχαία από κάθε batch .

Διαδικασία Εκπαίδευσης. Όλα τα μοντέλα εκπαιδεύτηκαν για 20 εποχές χρησιμοποιώντας learning rate scheduler και τον Adam optimizer[56]. Ο learning rate scheduler περιλαμβάνει δυο φάσεις [3]: μια αρχική γραμμική προθέρμανση μέχρι $lr = 0.001$, ακολουθούμενη από εκθετική μείωση του lr. Αυτή η προσέγγιση στοχεύει να βελτιώσει τη σύγκλιση κατά την εκπαίδευση.

1.4.7 Αξιολόγηση

Η αξιολόγηση πραγματοποιήθηκε χρησιμοποιώντας δύο μεθόδους: (*i*) linear probing και (*ii*) εκπαίδευση μιας κεφαλής ταξινόμησης MLP πάνω από τις παγωμένες αναπαραστάσεις OAcE . Ο στόχος της πειραματικής αξιολόγησης είναι να δοκιμάσει εάν ο κωδικοποιητής OAcE μπορεί να ενισχύσει δύο κωδικοποιητές εικόνας: CLIP και Image MAE. Είναι σημαντικό να σημειωθεί ότι οι αναπαραστάσεις CLIP δεν έχουν εκπαιδευτεί στη βάση δεδομένων που χρησιμοποιήθηκε για αξιολόγηση, ενώ στο Image MAE έχει πραγματοποιηθεί fine-tuning σε αυτή τη βάση δεδομένων. Οι αξιολογημένες μέθοδοι είναι οι εξής:

1. **GT**: Το ground truth αποτέλεσμα προέκυψε εκπαιδεύοντας τους ταξινομητές στις αναπαραστάσεις του δασκάλου. Είναι σαν τα μοντέλα να έχουν πρόσβαση στις ¨τέλειες¨ αναμνήσεις των ενεργειών που σχετίζονται με κάθε αντικείμενο. Αυτό αναδεικνύει το χρήσιμο σήμα στις αναπαραστάσεις του μοντέλου του δασκάλου.
2. **OAcE σε CLIP**: Εκπαίδευση ταξινομητών στις αναπαραστάσεις OAcE , με το CLIP ως κωδικοποιητή εικόνας.
3. **CLIP**: Εκπαίδευση ταξινομητών στις αναπαραστάσεις CLIP .
4. **OAcE σε IMAE**: Εκπαίδευση ταξινομητών στις αναπαραστάσεις OAcE , με το Image MAE ως κωδικοποιητή εικόνας.
5. **IMAE**: Εκπαίδευση ταξινομητών στις αναπαραστάσεις Image MAE .
6. **OAcE + IMAE**: Εκπαίδευση ταξινομητών στις συγχωνευμένες αναπαραστάσεις του Image MAE και του OAcE.

Linear probing. Το linear probing έχει χρησιμοποιηθεί ως πρωτόκολλο αξιολόγησης αναπαραστάσεων σε διάφορες μελέτες, συμπεριλαμβανομένων των [86, 45]. Περιλαμβάνει την εκπαίδευση ενός γραμμικού ταξινομητή πάνω από τις αναπαραστάσεις. Στο πλαίσιο του συνόλου δεδομένων Something's Affordance, η multi-label ταξινόμηση απαιτεί την εκπαίδευση πέντε δυαδικών γραμμικών ταξινομητών - έναν για κάθε κατηγορία προσφερόμενης δυνατότητας. Ο ταξινομητής που επιλέχθηκε για αυτό το πειραματικό τμήμα ήταν η Λογιστική Παλινδρόμηση, η οποία είναι ένα γενικευμένο γραμμικό μοντέλο. Τα αποτελέσματα παρουσιάζονται στους Πίνακες 3 και 4.

Configuration	Recall	Precision	F1 Score	Accuracy
GT	0.7508	0.9444	0.8349	0.8774
OAcE on CLIP	**0.7275**	**0.9224**	**0.8116**	**0.8610**
CLIP	0.7217	0.9147	0.8050	0.8561
OAcE on IMAE	0.6514	0.8986	0.7512	0.8256
IMAE	0.6863	0.8942	0.7740	0.8364
OAcE + IMAE	0.6964	0.8973	0.7821	0.8411

Table 3: Linear Probing μετρικές απόδοσης για τον διαχωρισμό βάσει βίντεο του συνόλου δεδομένων Something's Affordance

Κεφαλή Ταξινόμησης MLP. Το linear probing είναι ένα χρήσιμο πρωτόκολλο αξιολόγησης, δεν μπορεί να εκμεταλλευτεί μη γραμμικές αναπαραστάσεις. Μια μικρής κλίμακα κεφαλή MLP εκπαι-

Configuration	Recall	Precision	F1 Score	Accuracy
GT	0.6256	0.6240	0.5543	0.6845
OAcE on CLIP	**0.6360**	**0.6404**	**0.5707**	**0.6980**
CLIP	0.6256	0.6240	0.5543	0.6845
OAcE on IMAE	0.5575	0.5853	0.4821	0.6341
OAcE + IMAE	0.5994	0.5931	0.5341	0.6722
IMAE	0.5984	0.5907	0.5301	0.6681

Table 4: Linear Probing μετρικές απόδοσης για τον διαχωρισμό βάσει αντικειμένου του συνόλου δεδομένων Something's Affordance

δεύτηκε για την αξιολόγηση προς αυτή την κατεύθυνση. Η αρχιτεκτονική της κεφαλής ταξινόμησης είναι η εξής:

- Γραμμικό στρώμα (εισόδου d_t = 384, εξόδου = 1024)
- Στρώμα ενεργοποίησης Ρελυ
- Γραμμικό στρώμα (εισόδου: d_t = 1024, εξόδου = 5)
- Σιγμοειδής Ενεργοποίηση σε κάθε έξοδο

Η εκπαίδευση του νευρωνικού δικτύου ακολούθησε μια παρόμοια προσέγγιση με το μοντέλο OAcE Mapper, χρησιμοποιώντας τον αλγόριθμο Adam [56] και έναν learning rate scheduler με μέγιστη ταχύτητα μάθησης $0,001$. Για την τελική ταξινόμηση, εφαρμόζεται χρήση κατωφλίου (thresholding) στις εξόδους του τελευταίου στρώματος του ταξινομητή, οι οποίες βρίσκονται εντός του διαστήματος $[0, 1]$ λόγω της σιγμοειδούς ενεργοποίησης. Το κατώφλι ρυθμίζεται στο validation set, σε κάθε μια από τις πέντε κεφαλές ξεχωριστά, για να μεγιστοποιήσει το σκορ F1 του ταξινομητή. Τα πειραματικά αποτελέσματα παρουσιάζονται στους Πίνακες 5 και 6.

Configuration	Recall	Precision	F1 Score	Accuracy
GT	0.8265	0.9380	0.8776	0.9045
OAcE on CLIP	**0.8467**	**0.8782**	**0.8611**	**0.8878**
CLIP	0.8195	0.8858	0.8505	0.8817
OAcE on IMAE	0.8138	0.8173	0.8145	0.8493
AcE + IMAE	0.8051	0.8331	0.8174	0.8538
IMAE	0.7785	0.8359	0.8046	0.8458

Table 5: Μετρικές απόδοσης της MLP ταξινόμησης για τον διαχωρισμό βάσει βίντεο του συνόλου δεδομένων Something's Affordance

Configuration	Recall	Precision	F1 Score	Accuracy
GT	0.7508	0.9444	0.8349	0.8774
OAcE on CLIP	**0.6870**	**0.6834**	**0.6723**	**0.7820**
CLIP	0.6840	0.6742	0.6598	0.7720
AcE on IMAE	0.5271	0.6574	0.5656	0.7136
AcE + IMAE	0.5501	0.6656	0.5838	0.7218
IMAE	0.5410	0.6633	0.5763	0.7186

Table 6: Μετρικές απόδοσης της MLP ταξινόμησης για τον διαχωρισμό βάσει αντικειμένων του συνόλου δεδομένων Something's Affordance

1.4.8 Συμπεράσματα και μελλοντικές κατευθύνσεις

Γενικά, οι αναπαραστάσεις του Video MAE παρουσιάζουν καλύτερη απόδοση σε σύγκριση με τους κωδικοποιητές εικόνας. Αυτή η βελτίωση θα μπορούσε να οφείλεται στο ότι οι κωδικοποιητές εικόνας εκπαιδεύονται σε δεδομένα εκτός του πεδίου, ενώ το Video MAE έχει εκπαιδευθεί στο σύνολο δεδομένων SSv2. Για να το διερευνήσουμε αυτό πραγματοποιούμε fine-tuning του Image MAE σε εικόνες από το σύνολο δεδομένων. Ωστόσο, στα πειράματα ο κωδικοποιητής CLIP εξακολουθεί να υπερτερεί του Image MAE και ένα πιο οριστικό αποτέλεσμα θα απαιτούσε και το fine-tuning ενός μοντέλου CLIP, χρησιμοποιώντας το σύνολο δεδομένων SSv2.

Ωστόσο, θεωρούμε το γεγονός ότι οι αναπαραστάσεις Video MAE παρουσιάζουν καλύτερη απόδοση ως ένδειξη ότι υπάρχει χρήσιμο σήμα στις αναπαραστάσεις αυτές και τα πειράματά μας παρουσιάζουν μια προσπάθεια να το αξιοποιήσουμε. Οι Video MAE αναπαραστάσεις φαίνεται να έχουν σημαντικά μη γραμμικά χαρακτηριστικά, καθώς η απόδοσή τους βελτιώνεται σημαντικά στην ταξινόμηση με χρήση MLP. Γενικά, η προτεινόμενη μέθοδος OAcE παρέχει μια μικρή βελτίωση στους κωδικοποιητές εικόνας. Αυτή η βελτίωση είναι πιο εμφανής στον διαχωρισμό του συνόλου δεδομένων με βάση τα αντικείμενα. Αυτός ο διαχωρισμός με βάση τα αντικείμενα παρουσιάζει μια μεγαλύτερη πρόκληση για τα μοντέλα, καθώς εισάγει άγνωστα αντικείμενα test set.

Συνολικά, το OAcE με CLIP παρουσιάζει καλύτερη απόδοση, πλησιάζοντας τις Video MAE. Στην περίπτωση του Image MAE, το OAcE δεν παρείχε πάντα βελτιώσεις από μόνο του. Συνοψίζοντας, οι μεθόδοι που δοκιμάστηκαν παρουσιάζουν μία περιορισμένη αποτελεσματικότητα, ωστόσο ενδέχεται να χρειάζονται τροποποιήσεις στις μεθόδους ή μεγαλύτερα σύνολα δεδομένων για να πραγματοποιηθεί χρήσιμη μεταφορά πληροφορίας από τη δράση στο αντικείμενο.

Περιορισμοί της μεθόδου αξιολόγησης. Η τρέχουσα αξιολόγηση περιορίζεται σε ένα μικρό σύνολο δεδομένων με λίγες κατηγορίες δράσεων. Σε μελλοντικές έρευνες, περισσότερες κατηγορίες δυνατοτήτων θα μπορούσαν να εξαχθούν από αυτό το σύνολο δεδομένων. Μια πιο ολοκληρωμένη αξιολόγηση θα περιελάμβανε τη χρήση μεγαλύτερων συνόλων δεδομένων όπως το Ego4D [36] και το EPIC-Kitchens[17]. Ωστόσο, μια σημαντική πρόκληση θα ήταν η εκπαίδευση του Video MAE ViT, ή ενός εναλλακτικού μοντέλου δασκάλου σε αυτά τα μεγαλύτερα σύνολα δεδομένων, λόγω των μεγάλων χρονικών διαρκειών και της υψηλότερης ανάλυσης των βίντεο.

Επιπλέον, όπως σημειώθηκε προηγουμένως, στα πειράματά μας ο κωδικοποιητής εικόνας CLIP υπερτερεί του Image MAE, παρόλο που ο κωδικοποιητής CLIP εκπαιδεύεται μόνο σε δεδομένα εκτός του πεδίου (out-of-domain). Για να ενισχυθεί το επιχείρημα για τη μέθοδο αναπαράστασης δράσης-σε-αντικείμενο, είναι απαραίτητο να πραγματοποιηθούν πειράματα και με κάποιο CLIP μοντέλο εκπαιδευμένο σε εικόνες από το ίδιο σύνολο. Ωστόσο, δεδομένου ότι δεν είναι διαθέσιμος ο επίσημος κώδικας για την εκπαίδευση του CLIP , η διαδικασία αυτή αναβάλλεται για μελλοντική διερεύνηση.

Περιορισμοί της αρχιτεκτονικής μοντέλου. Ένας από τους περιορισμούς του OAcE είναι η εξάρτησή του από ένα μοντέλο ανίχνευσης αντικειμένων (π.χ. YOLO [89], SAM [57], EgoHOS [123], Mask R-CNN [44]) για την εξαγωγή των αντικειμένων από τα βίντεο. Για τα πειράματα αυτού του μέρους αποφασίστηκε να χρησιμοποιηθούν τα bounding boxes του συνόλου Something-Else και στη συνέχεια να διερευνηθεί μία μέθοδος εκμάθησης αναπαραστάσεων που εξάγει αυτόματα αντικείμενα και αναπαραστάσεις από μια σκηνή. Αυτή η μέθοδος είναι το Slot Attention, και το επόμενο κεφάλαιο καταγράφει μια προσπάθεια κατανόησης των κύριων ιδεών της και να αξιολογήσει τις αναπαραστάσεις που προκύπτουν.

1.5 Αναπαραστάσεις Slot Attention

1.5.1 Θεωρητικό Υπόβαθρο

Slot Attention. Αυτό το κεφάλαιο επικεντρώνεται στην αντικειμενοκεντρική μέθοδο Slot Attention που μπορεί να διαχωρίσει αυτόματα μια εικόνα ή ένα βίντεο σε αντικείμενα. Σε αυτό το πείραμα, οι αναπαραστάσεις αντικειμένων προέρχονται από το μοντέλο SOLV [3]. Αυτό το μοντέλο επιτυγχάνει επιτυχώς τον διαχωρισμό πολλαπλών αντικειμένων σε βίντεο εξάγοντας αναπαραστάσεις για κάθε ένα από τα αντικείμενα. Χρησιμοποιώντας το σύνολο δεδομένων Something's Affordances με τον διαχωρισμό με βάση βίντεο, στοχεύουμε να αξιολογήσουμε αυτές τις αναπαραστάσεις στην εργασία της κατηγοριοποίησης προσφερόμενων δυνατοτήτων. Πριν από την παρουσίαση της μεθοδολογίας και των πειραματικών αποτελεσμάτων, παρουσιάζονται ορισμένα από τα πιο σημαντικά στοιχεία του μοντέλου.

Η μέθοδος Slot Attention[67], είναι μια αρχιτεκτονική βασισμένη στη μέθοδο attention. Ο στόχος της είναι να συνδέσει αντικείμενα από μια οπτική είσοδο σε ένα σύνολο από υποδοχές (slots). Πιο συγκεκριμένα, η μέθοδος αυτή δέχεται μια εικόνα εισόδου που έχει διαιρεθεί και κωδικοποιηθεί σε N διανύσματα χαρακτηριστικών με κωδικοποίηση θέσης και τα επεξεργάζεται για να παράγει K διανύσματα υποδοχών. Τα διανύσματα υποδοχών μπορεί να αρχικοποιούνται τυχαία ή να είναι παράμετροι προς εκμάθηση, όπως στο SOLV. Αυτή η μέθοδος επιτρέπει σε κάθε υποδοχή να εξειδικεύεται σε έναν συγκεκριμένο γενικευμένο τύπο αντικειμένου.

Η μέθοδος αυτή συνήθως χρησιμοποιείται σε υποπλήρεις αυτο-κωδικοποιητές και τα διανύσματα υποδοχών τροφοδοτούνται σε έναν από-κωδικοποιητή Spatial Broadcast Decoder [113], όπου ανακατασκευάζει την αρχική είσοδο, είτε σε επίπεδο pixel, είτε σε επίπεδο αναπαραστάσεων. Έτσι, η εκπαίδευση γίνεται με χρήση της απώλειας ανακατασκευής (reconstruction loss).

Invariant Slot Attention. Η αρχιτεκτονική Invariant Slot Attention (ISM)[8] επιδιώκει την επεξεργασία του οπτικού σήματος με τρόπο που διαχωρίζει την εμφάνιση του αντικειμένου από τη στάση (pose) του αντικειμένου (θέση, προσανατολισμός και κλίμακα). Η ISM εφαρμόζει κωδικοποίηση θέσης στα διανύσματα χαρακτηριστικών των tokens με βάση το σχετικό πλαίσιο αναφοράς κάθε υποδοχής. Η μέθοδος ISA μπορεί να συνδυάζει αμεταβλητότητα ως προς τις τρεις ιδιότητες της στάσης ενός αντικειμένου: μετατόπιση, κλίμακα και περιστροφή. Τα καλύτερα αποτελέσματα επιτυγχάνει το μοντέλο που εισάγει την αμεταβλητότητα ως προς τη θέση και την κλίμακα: Translation and Scaling Invariant Slot Attention (ISA-TS) και αυτό χρησιμοποιείται στο SOLV.

Self-supervised Object-centric Learning for Videos (SOLV)[3] . Ο στόχος αυτού του μοντέλου είναι να διαχωρίζει βίντεο του πραγματικού κόσμου σε αντικείμενα. Η μέθοδος SOLV (Εικόνα 5) το επιτυγχάνει αυτό εφαρμόζοντας χωρο-χρονική (spatial-temporal) Slot Attention. Αρχικά, κάθε frame περνάει από χωρικό Slot Attention τύπου ISM, όπου υπολογίζονται τα αντικείμενα και οι αναπαραστάσεις τους. Στη συνέχεια, κάθε υποδοχή ενισχύεται με χρονική πληροφορία δίνοντας προσοχή στις αντίστοιχες υποδοχές σε γειτονικά frame. Το μοντέλο εκπαιδεύεται ως αυτό-κωδικοποιητής με απόκρυψη, ανασυνθέτοντας το κεντρικό frame του βίντεο σε επίπεδο χαρακτηριστικών, προερχόμενο από τον κωδικοποιητή DINOv2 [75].

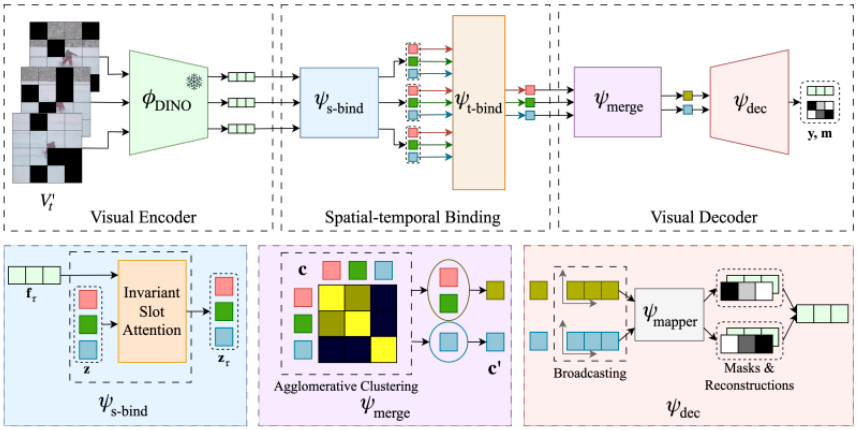
Ο κωδικοποιητής DINOv2 [75]είναι το πρώτο υποσύστημα στην αλυσίδα επεξεργασίας του SOLV. Λαμβάνει ως είσοδο ένα βίντεο με $2n+1$ frame, χωρίζει κάθε frame σε $N = HW/P^2$ μη επικαλυπτόμενα token μεγέθους P (Εξίσωση 71), εφαρμόζει απόκρυψη σε κάποια από τα token κατά τη διάρκεια της εκπαίδευσης) και κωδικοποιεί κάθε token. Το επόμενο υποσύστημα είναι το Spatial Binder, το οποίο εφαρμόζει ISA-TS σε κάθε frame ανεξάρτητα. Αυτό παράγει $2n+1 \times K$ διανύσματα υποδοχών.

Τα αρχικά διανύσματα υποδοχών είναι παράμετροι προς εκμάθηση. Δεδομένου ότι τα γειτονικά frame έχουν παρόμοια οπτική πληροφορία και τα απεικονιζόμενα αντικείμενα δεν αλλάζουν

δραστικά από το ένα στο επόμενο, θεωρούμε ότι στις περισσότερες περιπτώσεις οι υποδοχές με τον ίδιο δείκτη θα συνδέονται με τα ίδια αντικείμενα σε όλα τα frame. Έτσι το υποσύστημα Temporal Binder που ακολουθεί είναι ένας κωδικοποιητής μορφής transformer, ο οποίος ενισχύει τις αναπαραστάσεις των υποδοχών με πληροφορία από τις υποδοχές από τα υπόλοιπα frame . Για κάθε υποδοχή, η μονάδα Self-attention επεξεργάζεται τα $2n + 1$ διανύσματα υποδοχών, παράγοντας ένα τελικό διάνυσμα υποδοχής που περιέχει χρονική πληροφορία. Στα διανύσματα υποδοχών του κάθε frame έχει προστεθεί κωδικοποίηση χρονικής θέσης (temporal positional encoding), για να αξιοποιηθεί το σήμα χρονικής αιτιότητας που είναι διαθέσιμο στα δεδομένα βίντεο.

Στη συνέχεια, το υποσύστημα Slot Merger υπολογίζει δυναμικά τον βέλτιστο αριθμό υποδοχών για κάθε εικόνα και ομαδοποιεί τα διανύσματα χρησιμοποιώντας τον αλγόριθμο Agglomerative Clustering (AC). Τέλος, ένας Spatial Broadcast Decoder [113] λαμβάνει τον μειωμένο αριθμό διανυσμάτων και ανακατασκευάζει τα χαρακτηριστικά του κεντρικού frame, με βάση το οποίο υπολογίζεται η απώλεια ανακατασκευής (reconstruction loss).

Συνολικά, το SOLV εξάγει αναπαραστάσεις αντικειμένων ανακατασκευάζοντας το κεντρικό frame σε επίπεδο αναπαραστάσεων, ενώ χρησιμοποιεί πληροφορίες από ολόκληρο το βίντεο. Ενδιαφέρον παρουσιάζει το γεγονός ότι οι μάσκες τμηματοποίησης (segmentation masks) αντικειμένων προκύπτουν ως υποπροϊόν αυτής της αυτό-επιβλεπόμενης διαδικασίας. Σε αυτή την πειραματική ενότητα δοκιμάζουμε ένα άλλο υποπροϊόν αυτής της διαδικασίας — τις αναπαραστάσεις των υποδοχών — και τη χρησιμότητά τους για την κατηγοριοποίηση των προσφερόμενων δυνατοτήτων των αντικειμένων.

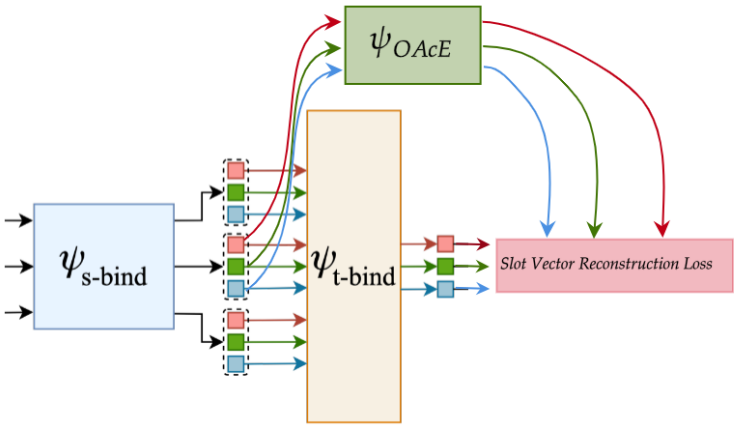


Σχήμα 5: Η αρχιτεκτονική του μοντέλου SOLV . Πηγή: [3]

1.5.2 Πειραματική Μέθοδος

Στην πειραματική ενότητα αυτή, αξιολογούμε τις αναπαραστάσεις υποδοχών του SOLV στην κατηγοριοποίηση των προσφερόμενων δυνατοτήτων των αντικειμένων, χρησιμοποιώντας το σύνολο δεδομένων Something's Affordances. Σε αντίθεση με τα πειράματα με τις αναπαραστάσεις του Video MAE όπου ο κωδικοποιητής OAcE δεχόταν ως είσοδο μία εικόνα αντικείμενου, το μοντέλο SOLV είναι ικανό να επεξεργάζεται ολόκληρη τη σκηνή και να την τμηματοποιεί αυτόματα.

Αρχικά, πραγματοποιήθηκε fine-tuning του μοντέλου SOLV με βίντεο από το train set της διαίρεσης βάσει βίντεο του Something's Affordances για 100 εποχές. Ο κώδικας εκπαίδευσης παρέχεται από το συμπληρωματικό υλικό του [3]. Λόγω της πολυπλοκότητας αυτής της εκπαίδευσης,



Σχήμα 6: Η εκπαίδευση του OAcE στο Spatial-temporal Binder του μοντέλου SOLV. Προσαρμόστηκε από: [3]

επικεντρωνόμαστε αποκλειστικά στη διαίρεση βάσει βίντεο, SA-vb, του συνόλου δεδομένων και η μελέτη στη διαίρεση βάσει αντικειμένων μετατίθεται σε μελλοντική έρευνα.

Ο αρθρωτός σχεδιασμός (modular design) του μοντέλου SOLV επιτρέπει την εξαγωγή αναπαραστάσεων με αντικειμενοκεντρική προσέγγιση σε διάφορα επίπεδα ροής της πληροφορίας. Τα δύο σημεία εστίασης των πειραμάτων είναι οι διανυσματικές έξοδοι των Spatial Binder και του Temporal Binder. Είναι σημαντικό να σημειωθεί ότι, παρότι ο Spatial Binder επικεντρώνεται στα χωρικά χαρακτηριστικά σε επίπεδο frame, έχει εκπαιδευτεί ως μέρος ενός συνολικού συστήματος που επεξεργάζεται βίντεο. Τα διανύσματα του Spatial Binder βελτιστοποιούνται για να παρακολουθούν μέσω attention διανύσματα των γειτονικών frame και επομένως μπορούν να θεωρηθούν μέρος της ευρύτερης κατηγορίας μεθόδων μεταφοράς γνώσης από βίντεο (video-to-image knowledge distillation) σε εικόνα και από δράση σε αντικείμενο .

Παρομοίως με την προσέγγιση του προηγούμενου κεφαλαίου, επιχειρούμε να συσχετίσουμε κάποια πληροφορία σχετική με τις δράσεις με τα διανύσματα αναπαράστασης των αντικειμένων. Αυτό γίνεται με τη χρήση ενός MLP, το οποίο λαμβάνει τα διανύσματα υποδοχών από το κεντρικό frame ενός βίντεο και εκπαιδεύεται να προβλέπει τα διανύσματα που δημιουργούνται από το αποτέλεσμα του Temporal Binder, όπως φαίνεται στο Σχήμα 6. Αυτό το MLP εκπαιδεύεται σε ένα σύνολο δεδομένων από βίντεο δράσεων και ονομάζεται $OAcE_{SOLV}$. Το μοντέλο $OAcE_{SOLV}$ έχει την ακόλουθη αρχιτεκτονική:

- **Linear Layer 1**: `Linear(`D_{slot}`,` $4 \times D_{slot}$`)`
- **ReLU Activation**: `ReLU(inplace=True)`
- **Linear Layer 2**: `Linear(`$4 \times D_{slot}$`,` D_{slot}`)`
- **Dropout**: `nn.Dropout(p=0.1)`
- **Residual Connection**: `output += input`

Το train set του συνόλου δεδομένων αποτελείται από 62,330 βίντεο. Λόγω της αυξημένης πολυπλοκότητας του μοντέλου αυτού, σε κάθε εποχή λαμβάνεται ένα μικρότερο τυχαίο υποσύνολο μεγέθους 306 βίντεο, χωρίς επανατοποθέτηση. Η εκπαίδευση του $OAcE_{SOLV}$ διεξάγεται για 10 εποχές, χρησιμοποιώντας batches μεγέθους 18 και learning rate scheduling που περιλαμβάνει μία αρχική γραμμική προθέρμανση μέχρι να φτάσει $lr = 0.0004$, ακολουθούμενη από εκθετική

μείωση. Η συνάρτηση απώλειας που παρουσίασε τα καλύτερα αποτελέσματα ήταν η Smooth L1 Loss .

1.5.3 Αξιολόγηση

Για να αξιολογήσουμε τις αναπαραστάσεις των υποδοχών του SOLV, τις εκπαιδεύουμε στο επισημειωμένο σύνολο δεδομένων SA-vb, χρησιμοποιώντας τα bounding boxes της βάσης δεδομένων Something-Else. Η κεφαλή κατηγοριοποίησης προσφερόμενων δυνατοτήτων, (Affordance Categorization Module -ACM), είναι ένα MLP με την ακόλουθη αρχιτεκτονική:

- **Batch Normalization Layer** [50]: `BatchNorm1d(`D_{slot}`)`
- **Linear Layer 1**: `Linear(`D_{slot}`, 1024)`
- **ReLU Activation**: `ReLU(in place=True)`
- **Linear Layer 2**: `Linear(1024, 5)`
- **Dropout**: `Dropout(p=0.1)`
- **Sigmoid Activation**: `Sigmoid()`

Ο αλγόριθμος εκπαίδευσης του ACM (Αλγόριθμος 3) ξεκινά με την επεξεργασία των εικόνων του συνόλου δεδομένων μέσω του SOLV και τον υπολογισμό των αναπαραστάσεων υποδοχών και των αντίστοιχων attention maps. Στη συνέχεια, αυτές περνούν μέσω Slot Merger Module το οποίο συνδυάζει ορισμένες υποδοχές με βάση την ομοιότητά τους.

Οι attention maps χρησιμοποιούνται για την παραγωγή μασκών τμηματοποίησης (segmentation masks), αναθέτοντας το κάθε pixel στην υποδοχή με το μεγαλύτερο attention πάνω του. Στη συνέχεια εντοπίζεται η υποδοχή στην οποία έχουν ανατεθεί τα περισσότερα pixel εντός του bounding box του αντικειμένου. Τα διανύσματα αναπαραστάσεων των υποδοχών αυτών αντιστοιχίζονται με τις ετικέτες προσφερόμενων δυνατοτήτων, ενώ μία τυχαία επιλεγμένη υποδοχή από τις υπόλοιπες αντιστοιχίζεται σε ετικέτα αρνητικής κατηγοριοποίησης σε όλες τις προσφερόμενες δυνατότητες. Στη συνέχεια, το *ACM* εκπαιδεύεται χρησιμοποιώντας αυτά τα ζεύγη εισόδου-ετικέτας.

Η εκπαίδευση του ACM περιλαμβάνει εκπαίδευση για 20 εποχές, χρησιμοποιώντας batches μεγέθους 18 και learning rate scheduling που περιλαμβάνει αρχική γραμμική αύξηση μέχρι το $lr = 0.001$, ακολουθούμενη από εκθετική μείωση. Πάλι η συνάρτηση απώλειας που παρουσίασε τα καλύτερα αποτελέσματα ήταν η Smooth L1 Loss. Ποσοτικά αποτελέσματα παρουσιάζονται στον Πίνακα 23 και ποιοτικά αποτελέσματα στα Σχήματα 44 και 45. Στα ποιοτικά αποτελέσματα, οι μάσκες τμηματοποίησης των υποδοχών απεικονίζονται με διαφορετικά χρώματα και οι ετικέτες κατηγοριοποίησης τοποθετούνται στο κέντρο βάρους της μάσκας τμηματοποίησης κάθε υποδοχής.

Configuration	Recall	Precision	F1 Score	Accuracy
GT	0.7570	0.9407	0.8378	0.8793
$OAcE_{SOLV}$	**0.7109**	0.9470	**0.8103**	**0.8631**
SOLV Spatial	0.7065	0.9476	0.8076	0.8614

Table 7: Η απόδοση των αναπαραστάσεων υποδοχών του SOLV στο σύνολο δεδομένων *SA – Vb*

1.5.4 Παρατηρήσεις

Αρχικά, οι αναπαραστάσεις των υποδοχών παρουσιάζουν συγκρίσιμα αποτελέσματα με τις αναπαραστάσεις των μοντέλων του προηγούμενου κεφαλαίου. Αυτό συμβαίνει παρά το γεγονός ότι είναι μικρότερες σε μέγεθος ($D_{OAcE_{extSOLV}} = 128$, $D_{OAcE} = 384$) και πραγματοποιούν αυτόματη τμηματοποίηση, η οποία εισάγει κάποιο θόρυβο στη διαδικασία.

Τα αποτελέσματα δείχνουν ότι οι αναπαραστάσεις που προέρχονται από τον Temporal Binder του SOLV, μπορεί να περιέχουν κάποιο χρήσιμο σήμα που δεν υπάρχει στις αναπαραστάσεις του Spatial Binder. Το μοντέλο $OAcE_{\text{SOLV}}$ προσπαθεί να εκμεταλλευτεί αυτό το χρήσιμο σήμα και έχει ως αποτέλεσμα μια μικρή βελτίωση. Παρατηρούμε ότι η μετρική precision δεν βελτιώνεται στις αναπαραστάσεις GT και $OAcE_{\text{SOLV}}$. Αυτό πιθανότατα συμβαίνει επειδή το σύνολο δεδομένων περιέχει σημαντικά περισσότερες αρνητικές ετικέτες από θετικές, καθιστώντας προτιμότερο για τα μοντέλα να είναι συντηρητικά στις προβλέψεις τους. Ως αποτέλεσμα, η μετρική F1 προσφέρει μια πιο χρήσιμη εικόνα της απόδοσης του μοντέλου.

Τέλος, τα ποιοτικά αποτελέσματα δείχνουν ότι παρόλο που η διαδικασία εκπαίδευσης περιλαμβάνει ένα αντικείμενο ανά σκηνή, το μοντέλο μπορεί να ανιχνεύει και να κατηγοριοποιεί σωστά πολλαπλά αντικείμενα ανά σκηνή. Επιπλέον, σε ορισμένες περιπτώσεις, ένα μόνο αντικείμενο μπορεί να ανατίθεται σε πολλές υποδοχές. Αυτή η πιο λεπτομερής τμηματοποίηση μπορεί να είναι επιθυμητή σε ορισμένα σενάρια, αλλά όχι σε άλλα. Στις περισσότερες περιπτώσεις, όλες οι υποδοχές που αντιστοιχούν στο ίδιο αντικείμενο κατηγοριοποιούνται σωστά.

1.5.5 Συμπεράσματα και μελλοντικές κατευθύνσεις

Σε αυτή την ενότητα μελετήσαμε διάφορα μοντέλα που χρησιμοποιούν την αρχιτεκτονική Slot Attention και δοκιμάσαμε τις αναπαραστάσεις του μοντέλου SOLV στο σύνολο δεδομένων *SA – Vb*. Το μοντέλο SOLV είναι υψηλού ενδιαφέροντος για τη διπλωματική αυτή λόγω του αρθρωτού σχεδιασμού του, που επιτρέπει την εξαγωγή αντικειμενοκεντρικών αναπαραστάσεων από εικόνες και βίντεο.

Με τη χρήση των υποσυστημάτων Spatial και Temporal Binder, το μοντέλο SOLV έχει τη δυνατότητα να επεξεργαστεί ολόκληρες σκηνές, παρέχοντας αντικειμενοκεντρικές αναπαραστάσεις. Δείχνουμε ότι οι αναπαραστάσεις βίντεο από το Temporal Binder έχουν ένα μικρό πλεονέκτημα στην κατηγοριοποίηση δυνατοτήτων σε σύγκριση με τις στατικές αναπαραστάσεις εικόνων από το Spatial Binder. Πειραματιστήκαμε με μια παραλλαγή του Object Action-centric encoder, $OAcE_{\text{SOLV}}$, που επιχειρεί να συνδέσει κάποιες πληροφορίες του Temporal Binder με τις αναπαραστάσεις του Spatial Binder. Οι αναπαραστάσεις του $OAcE_{\text{SOLV}}$, επιτυγχάνουν μια μικρή βελτίωση. Επιπλέον, προκύπτουν θετικές ενδείξεις για την ικανότητα του μοντέλου για γενίκευση καθώς το μοντέλο κατηγοριοποιεί πολλαπλά αντικείμενα σε μια σκηνή, ενώ εκπαιδεύτηκε σε σκηνές με ένα επισημειωμένο αντικείμενο.

Περιορισμοί του συνόλου δεδομένων. Μια πιο ολοκληρωμένη αξιολόγηση θα περιλάμβανε τη χρήση μεγαλύτερων συνόλων δεδομένων, όπως το Ego4D [36] και το EPIC-Kitchens [17]. Επιπλέον, θα ήταν ενδιαφέρον να διερευνηθεί η ενσωμάτωση αυτών των αναπαραστάσεων σε αρχιτεκτονικές που στοχεύουν στην επίλυση προβλημάτων, όπως η Πρόβλεψη Δράσης (Action Anticipation)[127].

Αναπαραστάσεις για τον έλεγχο. Το πρόβλημα της κατηγοριοποίησης των προσφερόμενων δυνατοτήτων μπορεί να είναι πολύ χρήσιμο σε ρομποτικά συστήματα για σχεδιασμό μελλοντικών δράσεων. Ιδανικά, οι ίδιες αναπαραστάσεις θα πρέπει να είναι χρήσιμες στις εργασίες ρομποτικού ελέγχου. Παρόλα αυτά, μια αναπαράσταση που αποδίδει καλά σε προβλήματα αναγνώρισης δεν αποδίδει απαραίτητα καλά στις εργασίες ελέγχου [78]. Στο επόμενο κεφάλαιο, δοκιμάζουμε τις αναπαραστάσεις του SOLV σε ένα απλό πρόβλημα προσομοιωμένου ρομποτικού χειρισμού.

1.6 Αναπαραστάσεις Slot Attention για ρομποτικό έλεγχο

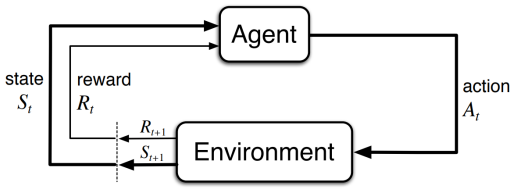
Σε αυτό το μέρος, μελετάμε μια μέθοδο για να συνδυάσουμε τις αναπαραστάσεις υποδοχών του μοντέλου SOLV, για τη δημιουργία αναπαραστάσεων εικόνων για ένα πρόβλημα προσομοιωμένου

ρομποτικού χειρισμού. Συγκεκριμένα, αυτή η μέθοδος εφαρμόζεται σε μια προσομοιωμένη ρομποτική εργασία από το Train Offline, Test Online (TOTO) benchmark [128]. Συγκρίνουμε την απόδοση του κωδικοποιητή εικόνας που βασίζεται στο SOLV σε με άλλους προ-εκπαιδευμένους κωδικοποιητές εικόνας. Τα αποτελέσματά δείχνουν ότι το SOLV γενικά επιτυγχάνει καλύτερη απόδοση σε αυτό το περιβάλλον. Παρόλα αυτά, τα αποτελέσματα είναι σε ένα προσομοιωμένο περιβάλλον που μπορεί να μην μεταφέρονται στον πραγματικό κόσμο και αυτό αποτελεί μία από τις βασικές προκλήσεις στην έρευνα της ρομποτικής [128]. Όμως, η αξιολόγηση στην προσομοιωμένη εργασία προσφέρει ένα πρώτο βήμα στη δοκιμή της μεθόδου μας πριν από τη μετάβαση σε πειράματα στον πραγματικό κόσμο (real-world testing).

Σχήμα 7: Η προσομοιωμένη εργασία ρομποτικής χειρισμού του TOTO[128]

1.6.1 Θεωρητικό Υπόβαθρο

Ενισχυτική Μάθηση. Σύμφωνα με τους Russell και Norvig [93], ένας πράκτορας είναι μια οντότητα που αλληλεπιδρά με ένα εξωτερικό περιβάλλον με σκοπό την επίτευξη ενός στόχου. Οι αλγόριθμοι Ενισχυτικής Μάθησης (Reinforcement Learning - RL) στοχεύουν στην ανάπτυξη πρακτόρων που αλληλεπιδρούν με ένα εξωτερικό περιβάλλον με τέτοιο τρόπο ώστε να μεγιστοποιούν το αναμενόμενο σήμα ανταμοιβής που λαμβάνουν από αυτό το περιβάλλον [104]. Αυτή η αλληλεπίδραση συνήθως μοντελοποιείται χρησιμοποιώντας *Μαρκοβιανές Διαδικασίες Αποφάσεων* (Markov Decision Process - MDPs): Ο πράκτορας αλληλεπιδρά με το περιβάλλον σε μια σειρά διακριτών χρονικών βημάτων, όπως φαίνεται στο Σχήμα 8.

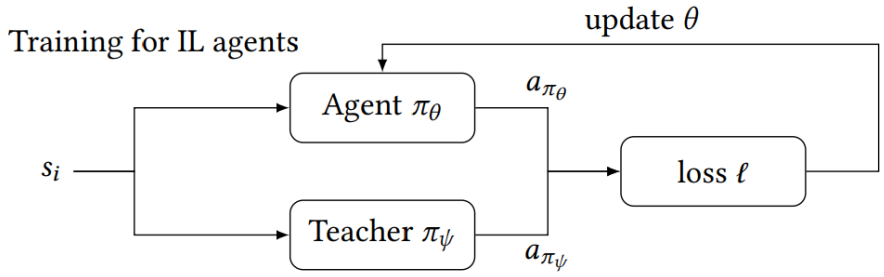


Σχήμα 8: Μαρκοβιανές Διαδικασίες Αποφάσεων. Πηγή: [104]

Σε κάθε χρονικό βήμα, ο πράκτορας λαμβάνει πληροφορίες σχετικά με την κατάσταση του περιβάλλοντος, $s_t \in S$, και επιλέγει μια δράση, $a_t \in A$. Αυτή η δράση, με τη σειρά της, επηρεάζει τη μετάβαση σε μια νέα κατάσταση, $s_{t+1} \in S$, καθώς και τη λήψη κάποιας ανταμοιβής, $r_{t+1} \in R$. Οι μεταβάσεις των καταστάσεων εξαρτώνται από τη δυναμική του συστήματος και τις πιθανότητες μετάβασης που συμβολίζονται ως $P : (S \times A)^2 \to [0, 1]$. Οι πιθανότητες μετάβασης $P(s', r|s, a)$

εκφράζουν την αβεβαιότητα της λήψης της ανταμοιβής r και της μετάβασης στην κατάσταση s' από την κατάσταση s όταν εκτελείται η δράση a. Η στρατηγική του πράκτορα για την επιλογή δράσεων εκφράζεται μέσω της συνάρτησης πολιτικής $\pi : S \to A$, η οποία αντιστοιχίζει καταστάσεις σε δράσεις [104].

Μάθηση μέσω Μίμησης (Imitation Learning). Σε αντίθεση με την ενισχυτική μάθηση όπου ο πράκτορας μαθαίνει αλληλεπιδρώντας με το περιβάλλον, στη μάθηση μέσω μίμησης, η εκμάθηση γίνεται μέσω κάποιου δασκάλου, που ο πράκτορας προσπαθεί να μιμηθεί. Με τη μέθοδο αυτή δεν χρειάζεται εξερεύνηση του περιβάλλοντος και αυτό είναι πολύ χρήσιμο σε περιπτώσεις που το κόστος και ο κίνδυνος πειραμάτων είναι υπερβολικά υψηλά, όπως στην αυτόνομη οδήγηση και στη ρομποτική [29].



Σχήμα 9: Μάθηση μέσω Μίμησης. Πηγή: [29]

Όπως και στην ενισχυτική μάθηση, η πολιτική είναι μία παραμετροποιημένη συνάρτηση που αντιστοιχίζει καταστάσεις σε δράσεις:

$$\pi_\vartheta : S \to A \tag{19}$$

Η διαδικασία εκμάθησης μια πολιτικής μέσω μίμησης παρουσιάζεται στο Σχήμα 9. Μια πολιτική έχει παραμέτρους ϑ, οι οποίες αντιπροσωπεύουν τις μεταβλητές που προσαρμόζονται κατά τη διάρκεια της μάθησης. Στις περισσότερες περιπτώσεις, ο αλγόριθμος δεν μπορεί να έχει άμεση πρόσβαση στην πολιτική του δασκάλου π_ψ, επειδή απαιτεί γνώση της εσωτερικής του κατάστασης, και έτσι η εκμάθηση γίνεται με χρήση παραδειγμάτων.

Behavioral Cloning. Η μέθοδος του Behavior Cloning είναι ένας από τους πρώτους και απλούστερους αλγορίθμους μάθησης μέσω μίμησης [29, 9]. Χρησιμοποιεί την τεχνική της επιβλεπόμενης μάθησης για να εκπαιδεύσει την πολιτική π_ϑ προβλέποντας την πιο πιθανή ενέργεια δεδομένης μιας κατάστασης, δηλαδή $\arg\max P(a|s)$, χρησιμοποιώντας σε ένα επισημειωμένο σύνολο δεδομένων που δημιουργήθηκε από τον δάσκαλο.

Ένα μειονέκτημα του Behavior Cloning είναι ότι σε πολύπλοκα προβλήματα, οι πολιτικές του δυσκολεύονται να γενικεύσουν. Αυτό συμβαίνει επειδή η πολιτική π_ϑ τείνει να αποτυγχάνει όταν συναντά καταστάσεις που δεν υπάρχουν στα παραδειγματα του δασκάλου. Ωστόσο, η τεχνική αυτή χρησιμοποιείται αποτελεσματικά για την εκκίνηση της εκπαίδευσης ενός πράκτορα πριν την εφαρμογή μιας μεθόδου Ενισχυτικής Μάθησης [29, 9].

Σε αυτή τη διπλωματική, εφαρμόζουμε Behavior Cloning σε μια απλή ρομποτική προσομοίωση για την αξιολόγηση μεθόδων εκμάθησης οπτικών αναπαραστάσεων.

1.6.2 Σύνολο δεδομένων

TOTO. Το TOTO [128] είναι ένα ρομποτικό benchmark που ανήκει σε μία προσπάθεια που γίνεται στον τομέα της ρομποτικής να αντιμετωπιστεί η έλλειψη τυποποίησης ανάμεσα στα ερευνητικά κέντρα. Το TOTO παρέχει πρόσβαση σε ρομποτικό εξοπλισμό και δεδομένα για offline εκπαίδευση. Το συνολο δεδομένων αποτελείται από ρομποτικές τροχιές που συλλέχθηκαν μέσω τηλεχειρισμού του ρομπότ, εμπλουτισμένες με θόρυβο και διαδρομές που δημιουργήθηκαν από πράκτορες εκπαιδευμένους μέσω Behavioral Cloning (BC). Το benchmark επικεντρώνεται σε δύο εργασίες χειρισμού: το άδειασμα υλικού από δοχείο σε δοχείο (pouring) και τη χρήση κουταλιού (scooping).

Το TOTO προσφέρει ένα πρωτόκολλο για την αξιολόγηση τόσο των οπτικών αναπαραστάσεων, όσο και των μεθόδων εκμάθησης πολιτικής. Σε αυτή την εργασία, επικεντρωνόμαστε αποκλειστικά στην αξιολόγηση των οπτικών αναπαραστάσεων, δοκιμάζοντάς τες με τη μέθοδο εκμάθησης πολιτικής BC, η οποία είναι μία μέθοδος *Μάθησης μέσω Μίμησης (Imitation Learning)*.

Προσομοίωση. Το TOTO περιλαμβάνει ένα περιβάλλον προσομοιώσεων για την εργασία έκχυσης και ένα σύνολο δεδομένων με 108 πορείες τηλεχειρισμού. Αυτή η προσομοίωση χρησιμοποιήθηκε για να αξιολογηθεί η μέθοδος αυτής της ενότητας. Η προσομοίωση προορίζεται για τις αρχικές δοκιμές στις μεθόδους μεθόδους τους και δεν αποτελεί μέρος του επίσημου πρωτοκόλλου αξιολόγησης TOTO. Όπως αναφέρθηκε νωρίτερα, τα αποτελέσματα της προσομοίωσης μπορεί να είναι παραπλανητικά και η υπερπροσαρμογή στο προσομοιωμένο περιβάλλον μπορεί να εμποδίσει τη γενίκευση στις πραγματικές συνθήκες. Παρόλα αυτά, η χρήση της προσομοίωσης αποτελεί ένα πολύτιμο αρχικό βήμα πριν από τη διεξαγωγή πειραμάτων στον πραγματικό κόσμο.

Η προσομοίωση χρησιμοποιεί το λογισμικό προσομοίωσης MuJoCo [108]. Ο προσομοιωμένος ρομποτικός βραχίονας είναι είναι τύπου Franka Emika Panda [41] με 7 βαθμούς ελευθερίας, με κάθε άρθρωση να περιορίζεται σε συγκεκριμένο εύρος θέσεων. Ο περιορισμός αυτός απλοποιεί λίγο το πρόβλημα, συρρικνώνοντας τον χώρο ελέγχου, αλλά γίνεται και για λόγους ασφαλείας σε πραγματικές εφαρμογές.

Όπως φαίνεται στο Σχήμα 7, ο ρομποτικός βραχίονας ξεκινάει κρατώντας ένα δοχείο γεμάτο με 12 μικρές σφαίρες. Ο στόχος είναι να αδειάσει όσο το δυνατόν περισσότερες σφαίρες σε ένα άλλο δοχείο. Οι αρχικές θέσεις των αρθρώσεων και η τοποθεσία του στόχου ποτηριού αρχικοποιούνται τυχαία για κάθε πείραμα. Μία δοκιμή θεωρείται επιτυχής αν τουλάχιστον μία σφαίρα καταλήξει στο δοχείο στόχος. Η μετρική ανταμοιβής είναι το ποσοστό των σφαιρών που έχουν κατατεθεί επιτυχώς στο δοχείο στόχος.

Το σύνολο εκπαίδευσης αποτελείται από τις 82 τροχιές που είναι επιτυχείς. Πριν από την εκπαίδευση, όλες οι εικόνες του συνόλου εκπαίδευσης κωδικοποιούνται με τον οπτικό κωδικοποιητή για να αξιολογηθούν. Επειδή ο οπτικός κωδικοποιητής είναι παγωμένος κατά τη διάρκεια της BC εκπαίδευσης, αυτό επιταχύνει την εκπαίδευση, καθώς διαφορετικά οι εικόνες θα έπρεπε να κωδικοποιηθούν ξανά για κάθε εποχή.

Κατά τη διάρκεια της εκπαίδευσης, ο αλγόριθμος BC εκπαιδεύει ένα νευρωνικό δίκτυο που λειτουργεί ως η πολιτική του πράκτορα. Το δίκτυο πολιτικής λαμβάνει ως είσοδο το διανύσμα οπτικής αναπαράστασης σε συνδυασμό με τις τρέχουσες γωνίες των αρθρώσεων του ρομπότ και παράγει τους στόχους για τις γωνίες των αρθρώσεων. Αυτές στη συνέχεια τροφοδοτούνται στον ελεγκτή MuJoCo, ο οποίος κινεί τον βραχίονα του ρομπότ. Η διάσταση εισόδου είναι $inp\_dim = R\_dim + 7$, όπου R_dim είναι η διάσταση του διανύσματος αναπαράστασης εικόνας. Η διάσταση εξόδου είναι $out\_dim = 7 \times h$, όπου h είναι ο ορίζοντας των ενεργειών που πρέπει να προβλέπονται κάθε φορά. Στα επόμενα πειράματα, $h = 10$. Η αρχιτεκτονική του δικτύου πολιτικής είναι η εξής:

- **Normalization (Input)**:

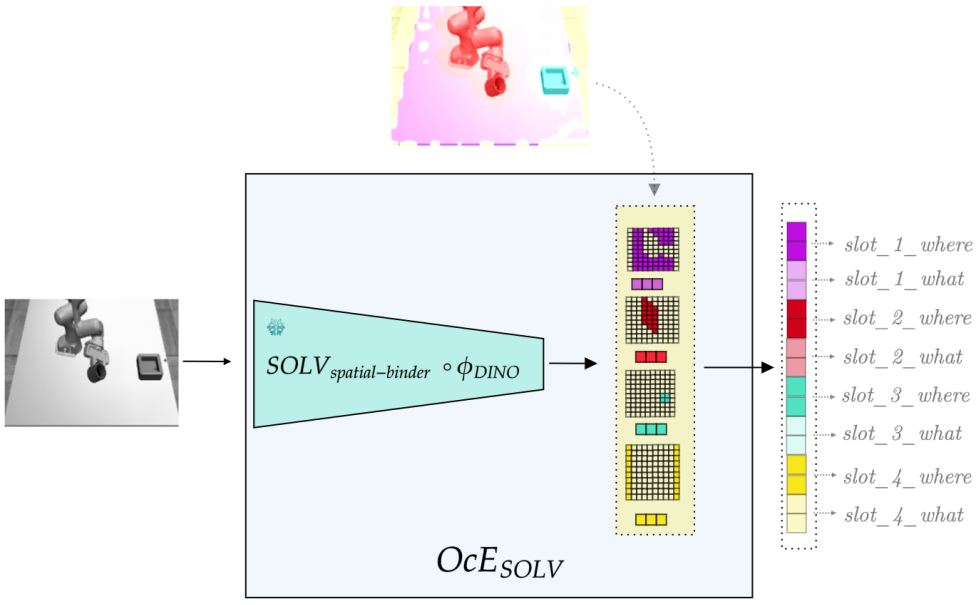
$$\text{norm\_output} = \frac{\text{Input} - \text{inp\_mean}}{\text{inp\_std}}$$

- **Linear Layer 1**: `Linear(inp_dim, hidden_dim)`
- **ReLU Activation**: `ReLU(inplace=True)`
- **Dropout**: `nn.Dropout(p=0.1)`
- **Linear Layer 2**: `Linear(hidden_dim, hidden_dim)`
- **ReLU Activation**: `ReLU(inplace=True)`
- **Dropout**: `nn.Dropout(p=0.1)`
- **Final Linear Layer**: `Linear(hidden_dim, out_dim)`
- **Rescale Output**:

$$\text{actions} = \text{out\_mean} + \text{out\_std} \times \text{final\_layer\_output}$$

Οι παράμετροι inp_mean, inp_std, out_mean, and out_std υπολογίζονται από το σύνολο εκπαίδευσης και αποθηκεύονται σε προσωρινές μεταβλητές εντός του μοντέλου. Η κανονικοποίηση αυτή κάνει πιο δίκαιη τη σύγκριση διαφορετικών κωδικοποιητών που παρέχουν εισόδους σε διάφορες μορφές. Επιπλέον, αντί να παράγει άμεσα το διάνυσμα δράσης, το δίκτυο προβλέπει πόσες τυπικές αποκλίσεις είναι η έξοδος από τον μέσο όρο των δράσεων. Αυτή η προσέγγιση μειώνει τη διακύμανση κατά την εκπαίδευση.

1.6.3 Πειραματική μέθοδος



Σχήμα 10: Ο κωδικοποιητής OcE_{SOLV}.

Σε αυτή την ενότητα, εισάγουμε μια μέθοδο για τον συνδυασμό των αντικειμενοκεντρικών αναπαραστάσεων του μοντέλου SOLV, για τη δημιουργία αναπαραστάσεων εικόνων για την εκπαίδευση

οπτικοκινητικών πολιτικών. Η προτεινόμενη μέθοδος είναι παρεμφερής με μεθόδους που χρησιμοποιούν αντικειμενοκεντρικές αναπαραστάσεις για εκμάθηση οπτικοκινητικών πολιτικών όπως το VIOLA (Visuomotor Imitation via Object-centric LeArning) [129]και το POCR (Pre-Trained Object-centric Representations) [99].

Η POCR πρώτα υπολογίζει το «πού» βρίσκονται τα αντικείμενα. Στη συνέχεια εφαρμόζει ένα μοντέλο τμηματοποίησης της εικόνας για την εξαγωγή μασκών για τα αντικείμενα. Οι συντεταγμένες αυτών των μασκών αποτελούν το «πού» του αντικειμενοκεντρικού διανύσματος αναπαράστασης. Στη συνέχεια υπολογίζεται η αναπαράσταση που αφορά το «τι» των αντικειμένων. Για κάθε μάσκα αντικειμένου, υπολογίζεται η αναπαράσταση του αντικειμένου με έναν προ-εκπαιδευμένο κωδικοποιητή εικόνας. Συνδυάζοντας τα διανύσματα «πού» και «τι» για κάθε αντικείμενο, η μέθοδος αυτή δημιουργεί αντικειμενοκεντρικά διανύσματα αναπαράστασης, τα οποία στη συνέχεια χρησιμοποιούνται στην εκμάθηση πολιτικής.

Με παρόμοιο τρόπο, χρησιμοποιούμε το προ-εκπαιδευμένο μοντέλο SOLV [3], το οποίο έγινε fine-tune στα βίντεο δράσεων του Something Something, για να παραγάγουμε έναν κωδικοποιητή εικόνων, που ονομάζουμε OcE_{SOLV}. Το πλεονέκτημα του SOLV είναι ότι μπορεί να παράγει ταυτόχρονα τόσο διανύσματα «τι» όσο και «πού» μέσω του υπολογισμού των διανυσμάτων των υποδοχών. Στη δική μας περίπτωση, η εκπαίδευση του TOTO απαιτεί ένα επίπεδο διάνυσμα για κάθε εικόνα και δεν περιλαμβάνει έναν κωδικοποιητή transformer ικανό να επεξεργαστεί πολλαπλά διανύσματα αντικειμένων. Για να δημιουργήσουμε ένα επίπεδο διάνυσμα αναπαράστασης για κάθε εικόνα, εξάγουμε έναν σταθερό αριθμό υποδοχών ανά εικόνα, παράγουμε τα διανύσματα «τι» και «πού» για κάθε αντικείμενο και τα συνδυάζουμε σε μία ενιαία αναπαράσταση.

Όπως συζητήθηκε στο προηγούμενο κεφάλαιο, η αρχιτεκτονική SOLV περιλαμβάνει μια μονάδα Slot Merger που συγχωνεύει υποδοχές με βάση την ομοιότητά τους. Σε αυτή την ενότητα, διαμορφώνουμε τον αλγόριθμο Agglomerative Clustering για να συγχωνεύσει τις αρχικές 8 υποδοχές σε 4.

Τα 4 διανύσματα υποδοχών, με διάσταση $D_{\text{what}} = 128$, αντιπροσωπεύουν τα διανύσματα «τι». Όπως αναφέρθηκε προηγουμένως, αν και το SOLV χρησιμοποιεί Invariant Slot Attention, τα διανύσματα υποδοχών εξακολουθούν να περιέχουν κάποια πληροφορία θέσης λόγω της κωδικοποίησης θέσης του κωδικοποιητή DIVOv2.

Βελτιώνουμε την απόδοση του κωδικοποιητή εικόνας εμπλουτίζοντας τις πληροφορίες για την θέση των αντικειμένων μέσω του attention της κάθε υποδοχής. Το attention mask κάθε υποδοχής έχει αρχικά σχήμα $h_{att} \times w_{att} = 24 \times 36$. Μειώνουμε το μέγεθός της σε $h'_{att} \times w'_{att} = 10 \times 10$ χρησιμοποιώντας bilinear interpolation. Στη συνέχεια, εφαρμόζουμε συνάρτηση softmax στις μάσκες προσοχής μειωμένου μεγέθους. Τα διανύσματα θέσης που προκύπτουν, με διάσταση $D_{\text{ωηερε}} = 100$, συνδυάζονται με τα διανύσματα υποδοχών για να παραχθεί μια επίπεδη αναπαράσταση για ολόκληρη την εικόνα με μέγεθος $D = 4 \times (128 + 100) = 912$.

1.6.4 Αξιολόγηση

Σε αυτή την ενότητα, εκπαιδεύσαμε πράκτορες με τη μέθοδο BC χρησιμοποιώντας διάφορους σύγχρονους κωδικοποιητές εικόνας: (i) BYOL [38], (ii) CLIP [86], (iii) DINOv2 [75], MoCo [46], Resnet50 [43] . Επιπλέον, αξιολογήσαμε τις αναπαραστάσεις του προ-εκπαιδευμένου μοντέλου SOLV που παρέχεται από [3], το οποίο εκπαιδεύτηκε στο σύνολο βίντεο Youtube-VIS 2019 [118]. Αυτή η αξιολόγηση είχε ως στόχο να ελέγξουμε εάν το fine-tuning του μοντέλου SOLV με βίντεο από το Something Something έχει θετική επίδραση. Ο κωδικοποιητής που βασίζεται στο πρώτο μοντέλο εμφανίζεται στα αποτελέσματα που ακολουθούν ως OcE_{SOLV_YT}.

Κάθε πράκτορας εκπαιδεύεται για 80 εποχές. Για να δώσουμε μια πιο ξεκάθαρη εικόνα, εκπαιδεύουμε 5 πράκτορες για κάθε κωδικοποιητή εικόνας και αξιολογούμε κάθε έναν σε 100 τυχαία αρχικοποιημένες τροχιές της ρομποτικής εργασίας. Τα αποτελέσματα παρουσιάζονται στον Πίνακα

8.

	Representation size	Success Rate	Mean Reward
BYOL	512	0.46 ± 0.05	17.48 ± 2.40
CLIP	512	0.49 ± 0.06	18.61 ± 4.09
DINOv2	768	0.55 ± 0.04	18.72 ± 1.78
OcE_{SOLV_SS}	912	**0.61** ± 0.04	**25.25** ± 2.95
OcE_{SOLV_YT}	912	0.46 ± 0.06	15.07 ± 2.24
MoCo	2048	0.31 ± 0.04	9.4 ± 2.16
ResNet50	2048	0.56 ± 0.11	21.15 ± 5.60

Πίνακας 8: Σύγκριση των προ-εκπαιδευμένων αναπαραστάσεων στον ρομποτικό χειρισμό του TOTO [128] .

Κάποιες παρατηρήσεις για τα παραπάνω αποτελέσματα:

- Το μοντλελο OcE_{SOLV} παράγει καλά αποτελέσματα, παρόλα αυτά και άλλοι κωδικοποιητές όπως DINOv2 και Resnet50, παράγουν πράκτορες με καλές επιδόσεις.
- Ένα μειονέκτημα της προτεινόμενης μεθόδου είναι ότι έχει αυξημένη υπολογιστική πολυπλοκότητα καθώς βασίζεται σε πολλά επίπεδα που βασίζονται στη μέθοδο attention .
- Το finetuning του μοντέλου SOLV σε βίντεο του συνόλου δεδομένων Something Something έχει θετική επίδραση στις αναπαραστάσεις.
- Στα αποτελέσματα πειραμάτων πραγματικού κόσμου που παρουσιάστηκαν στο [128], οι κωδικοποιητές εικόνας MoCo και BYOL εμφάνισαν καλύτερη απόδοση. Αυτό υπογραμμίζει την ασυνέπεια μεταξύ των αποτελεσμάτων της προσομοίωσης και των πραγματικών αποτελεσμάτων.

1.6.5 Συμπεράσματα και μελλοντικές κατευθύνσεις

Στην ενότητα αυτή, παρουσιάζουμε μια μέθοδο συνδυασμού των αναπαραστάσεων των υποδοχών του μοντέλου SOLV για την παραγωγή αναπαραστάσεων εικόνας για την εκμάθηση οπτικοκινητικών πολιτικών. Η προτεινόμενη μέθοδος εφαρμόζεται σε μια προσομοιωμένη εργασία ρομποτικού χειρισμού. Τα πειραματικά αποτελέσματα δείχνουν ότι ο κωδικοποιητής $OAcE_{SOLV}$ επιτυγχάνει καλά αποτελέσματα και ότι η διαδικασία finetuning σε βίντεο δράσεων βελτιώνει τις αναπαραστάσεις. Συμπερασματικά, η προτεινόμενη μέθοδος εμφανίζει ενθαρρυντικά αποτελέσματα για δοκιμή σε πιο σύνθετες ρομποτικές εργασίες και συνθήκες πραγματικού κόσμου.

Στο μέλλον θα ήταν ενδιαφέρον να επεκταθεί η μελέτη σε πιο πολύπλοκες εργασίες που απαιτούν σχεδιασμό σε περιβάλλοντα πολλαπλών αντικειμένων, όπως αυτά που παρέχονται στα περιβάλλοντα Franka Kitchen [39]και Meta-world [120].

Ο κωδικοποιητής εικόνας πάνω στον οποίο βασίζεται το SOLV είναι ο DINOv2[75]. Μια ενδιαφέρουσα μελλοντική κατεύθυνση θα μπορούσε να περιλαμβάνει τη μελέτη μιας αρχιτεκτονικής παρόμοιας με το SOLV που να εξάγει αναπαραστάσεις υποδοχών από οπτικές αναπαραστάσεις ειδικά εκπαιδευμένες για ρομπότικους χειρισμούς, όπως τα R3M [73] και LIV [69], που εκπαιδεύτηκαν σε μεγάλα σύνολα δεδομένων, όπως το Ego4D [36] και το EpicKitchen [17].

1.7 Συμπεράσματα και μελλοντικές κατευθύνσεις

Ο κύριος στόχος αυτής της διπλωματικής εργασίας ήταν η διερεύνηση μεθόδων για τη βελτίωση των αντικειμενοκεντρικών κωδικοποιητών εικόνας, εστιάζοντας σε μεθόδους που δημιουργούν συσχετίσεις αντικειμένων-δράσεων, βάσει δεδομένων που προέρχονται από βίντεο δράσεων. Αυτοί οι κωδικοποιητές εικόνας, που αναπτύχθηκαν μέσω οπτικής προ-εκπαίδευσης, προορίζονται για χρήση στα συστήματα αντίληψης ρομπότ και τεχνητών πρακτόρων.

Στην πρώτη πειραματική ενότητα διερευνήσαμε μια μέθοδο που στοχεύει στην κωδικοποίηση των εμπειριών δράσης, χρησιμοποιώντας έναν προ-εκπαιδευμένο Masked Auto-encoder για βίντεο, και τη συσχέτισή τους, μέσω Απόσταξης Γνώσης, με την απεικόνιση των σχετικών αντικειμένων. Προσπαθήσαμε να ενισχύσουμε δύο προ-εκπαιδευμένους κωδικοποιητές εικόνας: (ι) CLIP [86] και (ιι) Image MAE [45]. Αυτές οι αναπαραστάσεις αξιολογήθηκαν στην εργασία της *κατηγοριοποίησης προσφερόμενων δυνατοτήτων αντικειμένων*, χρησιμοποιώντας ένα σύνολο δεδομένων μικρής κίμακας, που δημιουργήσαμε χρησιμοποιώντας το σύνολο δεδομένων Something-Something v2 [35]. Τα πειράματα δείχνουν ότι οι μέθοδοι παράγουν μια μικρή αλλά σταθερή βελτίωση. Το κύριο μειονέκτημα αυτής της πρώτης μεθόδου είναι η εξάρτησή της από ένα σύστημα ανίχνευσης αντικειμένων. Συνεπώς, στη δεύτερη πειραματική ενότητα επικεντρωθήκαμε σε ένα μοντέλο βασισμένο στη μέθοδο Slot Attention [67] που εξάγει αυτόματα τα αντικείμενα.

Στη δεύτερη πειραματική ενότητα, οι αναπαραστάσεις αντικειμένων αντλήθηκαν από το μοντέλο *SOLV*[3], το οποίο επιτυγχάνει την τμηματοποίηση και εξαγωγή αναπαραστάσεων πολλαπλών αντικειμένων σε βίντεο. Το μοντέλο SOLV είναι υψηλού ενδιαφέροντος για τη διπλωματική αυτή, λόγω του αρθρωτού σχεδιασμού του, που επιτρέπει την εξαγωγή αντικειμενοκεντρικών αναπαραστάσεων από εικόνες και βίντεο. Χρησιμοποιήσαμε πάλι το ίδιο σύνολο δεδομένων *κατηγοριοποίησης δυνατοτήτων* για την αξιολόγηση των μεθόδων μας. Η μέθοδος παρουσιάζει ανταγωνιστικά αποτελέσματα, ενώ επιτυγχάνει επίσης αυτόματη τμηματοποίηση των εικόνων και σημαντική μείωση στο μέγεθος της αναπαράστασης ανά αντικείμενο. Επιπλέον, προέκυψαν θετικές ενδείξεις για την ικανότητα του μοντέλου για γενίκευση, καθώς το μοντέλο κατηγοριοποιεί πολλαπλά αντικείμενα σε μια σκηνή, ενώ εκπαιδεύτηκε σε σκηνές με ένα επισημειωμένο αντικείμενο.

Στην τρίτη πειραματική ενότητα, μελετήσαμε μια μέθοδο για να συνδυάσουμε τις αναπαραστάσεις υποδοχών του μοντέλου SOLV, για τη δημιουργία αναπαραστάσεων εικόνων για ένα πρόβλημα προσομοιωμένου ρομποτικού χειρισμού. Αξιολογούμε την απόδοση αυτού του κωδικοποιητή εικόνας έναντι άλλων προ-εκπαιδευμένων κωδικοποιητών εικόνας και τα αποτελέσματά μας δείχνουν ότι η μέθοδός μας επιτυγχάνει γενικά καλύτερη απόδοση. Η προτεινόμενη μέθοδος εμφανίζει ενθαρρυντικά αποτελέσματα για δοκιμή σε πιο σύνθετες ρομποτικές εργασίες και συνθήκες πραγματικού κόσμου.

Δημιουργώντας συσχετίσεις δράσης-αντικειμένου στις αναπαραστάσεις των κωδικοποιητών εικόνας, αυτή η διπλωματική επιδιώκει να συμβάλει στην ανάπτυξη πιο αποτελεσματικών συστημάτων όρασης για ρομπότ και τεχνητούς πράκτορες, επιτρέποντάς τους να κατανοούν καλύτερα τη σημασιολογία και τη δυναμική της αλληλεπίδρασης πράκτορα-αντικειμένου. Ως μελλοντική κατεύθυνση, μια πιο ολοκληρωμένη αξιολόγηση αυτών των μεθόδων θα μπορούσε να περιλαμβάνει μεγαλύτερα σύνολα δεδομένων όπως το Ego4D [36] και το EPIC-Kitchens[17], καθώς και περαιτέρω πειραματισμό με διαφορετικούς τρόπους μοντελοποίησης της χρήσιμης πληροφορίας που περιέχεται σε αυτά. Επιπλέον, οι μέθοδοι οπτικής προ-εκπαίδευσης για τη ρομποτική θα πρέπει να παρέχουν αναπαραστάσεις που να είναι χρήσιμες σε μια ποικιλία από εργασίες. Συνεπώς, θα ήταν χρήσιμο οι αντικειμενοκετρικές αναπαραστάσεις να αξιολογηθούν σε διάφορες εργασίες χειρισμού, σχεδιασμού και αναγνώρισης. Τέλος, θα ήταν ενδιαφέρον να μελετηθούν παρόμοιες μέθοδοι στο πλαίσιο της Συνεχούς (Continual) και Διαχρονικής (Lifelong) Μάθησης, και να εξερευνηθούν τεχνικές που επιτρέπουν στους πράκτορες να κωδικοποιούν και να δημιουργούν συσχετίσεις βάσει των δικών τους ενεργειών και εμπειριών.

2 Introduction

2.1 Motivation

A key objective of computer vision is to develop techniques that extract meaningful visual representations of the world. Robotics manipulation [6], planning [125], as well as human-robot interaction [2, 22, 23], are areas with many open problems being studied at the moment. The extraction of visual representations through visual pre-training methods (Figure 11) is promising because it reduces training time and improves performance and generalization, compared to end-to-end learning methods[52, 61, 95, 128]. These representations should be able to be utilized in a variety of downstream tasks and require minimal retraining [95, 70].

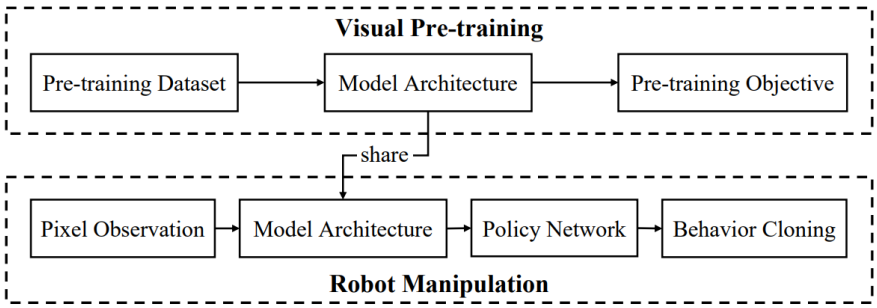


Figure 11: Visual Pre-Training for Robotics. Source: [52]

An emerging framework within visual pre-training methods is *object-centric representation learning*, where the goal is to represent complex environments in terms of objects, rather than treating the entire scene as a single entity. These methods are compatible with the way humans process visual signals by organizing them into objects [77] and show the potential to improve the generalization capabilities, explainability and sample efficiency of models [67, 8, 3].

Object-centric learning could draw inspiration from the field of psychology, where there has been extensive study in the way humans learn to interact with their environment by associating actions and words to objects. Experiments in developmental psychology show that infants first learn action-object associations, with word associations becoming more important later in development [24]. Developing robotic agents would potentially struggle to exactly follow human development due to limits in computational resources, datasets, or experimental constraints. However, this can inspire algorithms that seek to pre-train robotic perception systems through a form of curriculum learning [5, 100] to first focus on extracting representations from observed actions and then transition to learning from language-based supervision. This approach transitions from utilizing self-supervised learning on more easily accessible datasets, such as unlabeled videos, to the use of datasets that are annotated, which tend to be more costly. Additionally, it progresses from learning lower-order representations to higher-order representations.

Thesis Objective. This thesis focuses on ways that actions can be associated with objects, following the positive results of visual pre-training methods that focus on modeling action-centric information. Examples of this approach apply visual pre-training methods [87, 54, 61, 73, 69] using datasets that capture the way humans interact with objects [36, 17]. These datasets can be used to train the vision systems of agents and give them a head start in understanding the agent-object interaction dynamics of the real world.

Based on the above, the primary aim of this thesis is to explore methods for improving object-centric image encoders by focusing on methods that generate action-object associations based on knowledge sourced from videos of actions. The first task used to evaluate the effectiveness of these enhanced representations is affordance categorization, which is an appropriate assessment of action-centric representations. In the fields of Computer Vision, Robotics and Artificial Intelligence recognizing affordance can help systems anticipate and plan by providing information on possible interactions with objects and the environment. In addition, the effectiveness of these representations is assessed through a basic simulated robotic manipulation task.

2.2 Contributions

This thesis offers the following contributions.

1. **Something's Affordances: Curating a Small-Scale Affordance Categorization Dataset.** Something's Affordances is a small-scale dataset that extends the Something-Else dataset and focuses on affordance categorization. Its goal is to provide a proof of concept for the proposed methods which are aimed at enhancing image representations through the distillation of knowledge present in videos of actions. From the original dataset a small subset of action categories was selected based on their ability to test the representations. The multi-label affordance targets were extracted from the statistics of the dataset. The dataset offers a small-scale testing environment for simple versions of some of the methods, as a first step before scaling to bigger datasets with larger computational requirements.

2. **The Object Action-centric Encoder.** We have experimented with an action-to-object distillation process that transfers the knowledge of a pre-trained Video MAE to an image encoder. This framework attempts to encode (Video MAE) action experiences and associate them with the depiction of the interacting objects present in those experiences. Experiments show that the representations of Video MAE contain useful information that could be useful to the image encoders, and we test some methods to enrich them with this information. The methods tested show some limited capability but may need adjustments or larger datasets to effectively capture this information.

3. **SOLV [3] representations for affordance categorization**. We evaluate the image representations of objects using a model that utilizes the Slot Attention architecture. We utilize the model's modular design to extract image object-centric representations from images and we propose a method that attempts to associate some extra information about the actions with the object representation vectors. The model presents competitive results while also achieving automatic segmentation of the images and a substantial reduction in the per-object representation size. Furthermore, the model's ability to detect and categorize multiple objects in a scene, despite being trained with one object per scene, highlights its robustness and potential for generalization.

4. **SOLV [3] representations for control.** We introduce a method to combine the spatial slot representations of the SOLV model to generate image representations for a simulated robot manipulation task. We evaluate the performance of this SOLV-based image encoder against other pre-trained image encoders that were trained on out-of-domain data. Our results demonstrate that SOLV generally achieves better performance in this setting, although it comes at the cost of increased computational complexity.

3 Theoretical Background

This *Theoretical Background* section aims to present the theoretical foundations of this thesis by providing the necessary context and focusing on key topics relevant to the proposed methods and experiments. It draws insights from various sources, but mainly from:

- Christopher M. Bishop's *Pattern Recognition and Machine Learning* [7]
- Ian Goodfellow, Yoshua Bengio, and Aaron Courville's *Deep Learning* [32]
- Marco Gori's *Machine Learning: A Constraint-Based Approach* [33]
- Peter Norvig and Stuart J. Russell's *Artificial Intelligence: A Modern Approach* [93]
- Sergios Theodoridis' *Machine Learning: A Bayesian and Optimization Perspective* [107]
- Shai Shalev-Shwartz and Shai Ben-David's *Understanding Machine Learning: From Theory to Algorithms* [98]

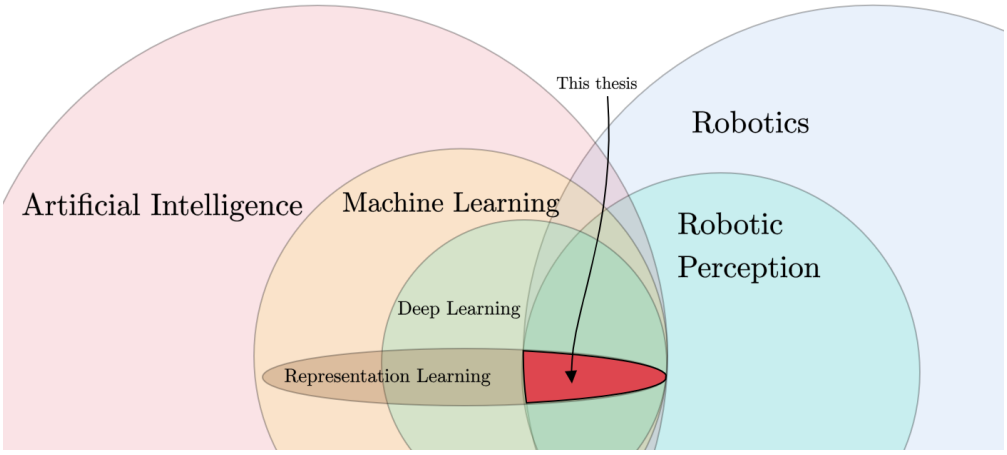


Figure 12: A Venn diagram illustrating the relationships between different fields relevant to this thesis.

3.1 Machine Learning

Machine Learning (ML) is a field of study focused on developing methodologies that enable computer programs to perform tasks by learning from data, instead of following explicit instructions. The ability of ML algorithms to learn from data without predefined instructions has proven highly effective in fields like computer vision and natural language processing, where humans and biological systems can perform complex tasks, but it is difficult to articulate the steps involved in accomplishing them. Furthermore, ML has seen success in tasks that are difficult or impossible for humans to perform, such as analyzing large amounts of data or making predictions based on complex patterns.

A concept that highlights the tasks in which ML excels at, is the symbolic vs. sub-symbolic dichotomy within the field of Artificial Intelligence (AI) [49, 33]. Symbolic AI is the branch of AI that focuses on methodologies that are highly dependent on the manipulation of symbols and attempt to approach tasks by programming computers to emulate human-like reasoning. These methods have the advantage of interpretability, as most of the reasoning process is transparent and understandable by humans. However, because symbolic systems use human-designed high-level representations, they often require significant human involvement and struggle in tasks that involve ambiguity, noisy data, or dynamic environments.

Notation	Description
$\mathcal{D}$	Domain set (set of all possible inputs)
$\mathcal{X}_{\text{train}}$	Training set
$\mathcal{X}_{\text{val}}$	Validation set
$\mathcal{X}_{\text{test}}$	Test set
$\mathcal{Z}$	Latent or internal representation space
$\mathcal{T}$	Target/decision space
$\mathcal{O}$	Final output space
$e : \mathcal{D} \to \mathcal{Z}$	Encoder function (maps domain inputs to latent space)
$f : \mathcal{Z} \to \mathcal{T}$	Decision function (maps latent space to target space)
$g : \mathcal{T} \to \mathcal{O}$	Post-processing function (maps decisions to final output)
$h_\partial = g \circ f \circ e$	Complete ML model
∂	Parameters of the ML model
$\mathcal{H}$	Hypothesis space
$p : \mathcal{O} \times \mathcal{O} \to \mathbb{R}$	Performance measure

Table 9: Machine Learning (ML) Notations

Computer vision tasks are mostly of sub-symbolic nature, because inputs at the pixel level have no semantic meaning and symbolic rules cannot easily be applied. Sub-symbolic AI, which includes most modern ML approaches, uses methods such as statistical learning and mathematical optimization to generate models from data. These models can be expressed as parameterized functions, and the learning process involves optimizing the functions' parameters using data. These methods do not have the interpretability of symbolic AI methods and this opaqueness has led to their characterization as *black box models*. This has led to the development of *explainable* ML, which focuses on techniques that provide explanations of the processes and results of ML models [71].

Producing useful ML models often requires a large number of parameters, which requires a large amount of data and computational resources for training. However, technological advances in hardware, such as GPUs and TPUs, and the availability of large datasets have helped overcome these bottlenecks, leading to significant advancements and furthering research in ML algorithms. Furthermore, in recent years, there is growing interest in hybrid approaches that combine both sub-symbolic and symbolic AI, that study methods that are not only effective but also more interpretable.

Formalizing Machine Learning Tasks and Models. Let us define a formalism for a generic ML task, model and learning, as presented in [33, 98]. The domain set , $\mathcal{D}$, represents the set of elements that an ML model, h_∂, is designed to process, producing outputs that belong to the output space, $\mathcal{O}$. For example, in the task of object classification, the domain set $\mathcal{D}$ consists of images of objects that we aim to classify into a predefined set of categories, $\mathcal{O}$, such as "cup", "chair", and others

An ML model can be formalized as a composition of three functions: $h_\partial = g \circ f \circ e$. Many ML models require a simpler formulation, in which some of the functions g, f, e are the identity functions. The encoder function $e : \mathcal{D} \to \mathcal{Z}$, maps the domain inputs to the latent space, also known as the internal representation space. The field that focuses on training such encoders is called representation learning. This topic will be explored further in a next chapter, as it constitutes a primary focus of this thesis. For example, the encoder can take images of objects, where each image $i \in \mathcal{D} = \mathbb{R}^{H \times W \times 3}$ (with height H, width W and 3 color channels) and encode

them into representation vectors $z = e(i) \in \mathcal{Z}$. The internal representation vectors aim to reduce the input size and transform the input into a useful form, preserving the most important information.

The function $f : \mathcal{Z} \to \mathcal{T}$ expresses the process that takes the internal representation $z \in \mathcal{Z}$, and produces a decision, $t = f(z) \in \mathcal{T}$, in the target space of the task. Finally, the post-processing function, $g : \mathcal{T} \to \mathcal{O}$, expresses the mapping of the decision to the final output, $o = g(t) \in \mathcal{O}$ in the space of the final task-specific output by applying activation functions, thresholding, or scaling.

In line with the previous example of the object classification task, the decision space can include vectors of assigned probabilities to each class. These probability vectors represent the model's confidence that the input belongs to each class. The post-processing function g could then apply transformations such as selecting the class with the highest probability or applying a threshold to produce a final result.

The class of models and the function, h_ϑ, is chosen from the set of all possible models $\mathcal{H}$. $\mathcal{H}$ is called the hypothesis space, as this choice represents the designer's hypothesis and prior knowledge about the task and domain set, which introduces certain restrictions to the learning process. These restrictions are known as *inductive bias* [98, 93].

Classification & Regression. If the target output is a continuous variable, the ML task is referred to as a *Regression* task. When the output is a finite number of categories, the task is called a *Classification* task. Classification into two categories is known as *Binary Classification*. Classification into three or more categories is refereed to as *Multi-class Classification*. When the goal is for each sample to be labeled with multiple, nonexclusive labels, the task is termed *Multi-label Classification*. An example of multi-label classification is *Affordance categorization*, a task that is within the scope of this thesis. For this task, each sample is an object, and the goal is to predict its nonexclusive affordances (e.g. a ball can be rollable and squeezable).

Datasets. In most of the tasks in which ML methods are used, the domain sets are infinite or finite but vast. For example, if $\mathcal{D}$ represents all RGB images of size 10×10, then a sample $i \in \mathcal{D}$ would be an element in $[0, 255]^{10\times10\times3}$, where each pixel has 3 color channels (red, green, blue) and intensity in the range $[0, 255]$. The cardinality of this domain set is $|\mathcal{D}| = 256^{300}$ [1]. It is clear that in most cases we cannot realistically train or test an ML model on the entire domain set, nor can we attempt to label every possible element.

ML methodologies are applied to subset datasets that are sampled from the domain set. The challenge is to develop models that learn from these datasets but can generalize to unseen data. This highlights the importance of sampling methods and the quality of the dataset. In many theoretical frameworks for ML, like probably approximately correct (PAC) learning [98], the assumption is that the sampling from the domain set takes place in an independently and identically distributed (i.i.d.) manner. However, this i.i.d. assumption is often not strictly true in practice. Understanding how this assumption is violated in real-world applications is important for developing effective ML models [122].

The datasets sampled from the domain set are usually split into three separate datasets: the train validation and the test set. Each of these sets performs a specific role in ML methods. The train set is used to optimize the parameters of the ML model according to the learning algorithm. The validation set is used to provide feedback during the training on how the model performs on unseen data. This feedback is used to adjust the training process' parameters, referred to as

[1] For comparison, recent studies have estimated the age of Universe is approximately 8×10^{17} seconds [40]

hyperparameters, without compromising the final evaluation. The test set is used for the final evaluation, providing an assessment of the model's performance on unseen data.

Overfitting & Underfitting. The main aim of ML methods is to produce models that perform well on unseen data drawn from the same domain set (or distribution), what is known as generalisability. Generalisability is measured by training a model based on performance metrics computed on the training set but evaluating it on the test set. In this context, the i.i.d assumption on the data generating process is important , as it allows statistical learning theory through the concept of *Empirical Risk Minimization* [98] to draw some conclusions on how the model's performance on the train set will affect its performance on the test set. [32].

Furthermore, a model's generalizability is affected by its *capacity*, which is its ability to fit complex datasets by approximating complex functions[32]. At first glance, the improved capacity may seem as an advantage, but this flexibility can lead to a phenomenon called *overfitting*, where the model fits the training set too tightly, diminishing its performance on unseen data. Underfitting is when low-capacity models underperform because the complexity of the task requires a greater amount of expressive power (Figure 13). In some ML models, like neural networks, the amount of overfitting is also affected by training hyperparameters like training duration, learning rate, etc. In recent years, there has been extensive research in finding methods that optimize the model's capacity and training methods to avoid over/underfitting and improve generalization.

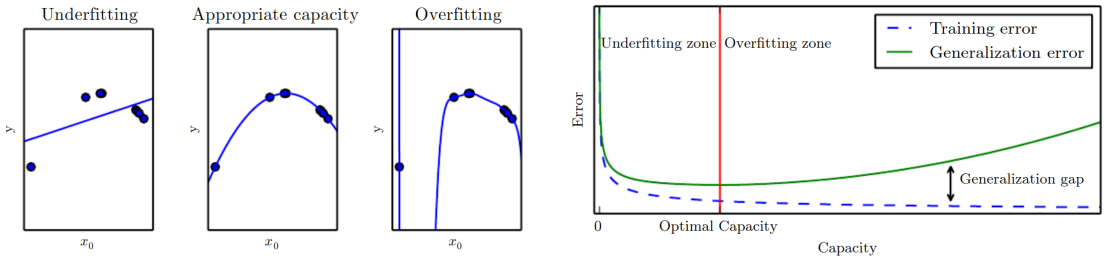


Figure 13: Model capacity, overfitting and underfitting. Source: [32]

Machine Learning Protocols. There are three main ML protocols that are determined by the different types of feedback that the models have access to during training.

- *Supervised Learning.* In this type of ML, the methods uses feedback in the form of labels. A labeled dataset includes a label for every sample, $\mathcal{X} = \{(x_1, y_1), \ldots, (x_m, y_m)\}$, where $x_i \in \mathcal{D}$ and $y_i \in \mathcal{O}$. The performance of the model is quantified using a performance measure function $p : \mathcal{O} \times \mathcal{O} \to \mathbb{R}$ that is used to compare the model predictions with the ground truth labels. During training a performance measure function is used to optimize the model parameters using the train set, $\mathcal{X}_{train}$. In the evaluation phase, performance measure functions are used on the labels of the test set, $\mathcal{X}_{test}$, and these functions can be different from the ones used in training.

- *Unsupervised Learning.* In Unsupervised Learning, there is no explicit feedback that informs the training process, and models attempt to discover patterns in datasets. Density estimation, clustering, and dimensionality reduction algorithms fall into this category.

- *Self-supervised Learning.* This ML approach, generally considered a subcategory of unsupervised learning, where the model generates its own supervision feedback from the data.

This technique is commonly used to pre-train encoder models that are then fine-tuned using supervised learning. Self-supervised methods using neural network encoder models fall within the scope of this thesis and are discussed in more detail in the next chapters.

- *Reinforcement Learning*. In Reinforcement Learning the models learn by interacting with an environment, and the feedback takes the form of a reward function. This type of ML is inspired by the way that humans learn and interact with their environments and has many applications in the field of robotics. RL is presented in greater detail in a following chapter.

Probabilistic methods. Probability theory is a common theoretical framework on which many ML methodologies are based on. It models the uncertainty and variability of the domain set, data generating process and model parametrization and provides a systematic approach handling noisy and incomplete data, incorporating prior knowledge to the learning process and updating the models using new information [107, 115, 32].

Using the probabilistic framework, the uncertainty in the datasets is quantified using probability distributions. Supervised learning can be seen as an attempt to estimate the conditional distribution $p(y|x)$ using parameterised functions by observing random variables x, y. This leads to two approaches for using the feedback of labels y: *generative* and *discriminative* learning. In *discriminative* learning the conditional distribution $p(y|x)$ is modeled directly, learning a mapping between input x and output y and focusing on the boundaries of the classes. On the other hand *generative* learning models the joint distribution $p(x, y)$ and uses the *product rule of probability* to make predictions (Equation 20). Generative models acquire a deeper understanding of the underlying distributions but usually require more complex parameterized functions and learning methods to achieve this.

$$P(y|x) = \frac{p(y,x)}{p(x)} = \frac{p(y,x)}{\sum_y p(y,x)}. \tag{20}$$

In the context of probabilistic models, Maximum Likelihood Estimation (MLE) and Maximum A Posteriori (MAP) are two common methods for estimating the model parameters. MLE assumes that the parameters ∂ are fixed but unknown and attempts to find the parameter values that maximize the likelihood of the observed data without incorporating any prior beliefs about the parameters. MAP estimation extends MLE by incorporating prior knowledge of the parameters through a prior distribution $p(\partial)$.

Performance Measures. *Permanence measures* are functions that quantify the model's performance and are an important part of ML theory and methods. For binary classification tasks, where each can either belong ($y_i = 1$), or not ($y_i = 0$), to a single category, some of these functions are based on the following numbers:

- *True Positives* (*TP*): The number of samples for which $y_i = 1$ and $h_\partial(x_i) = 1$
- *True Negatives*(*TN*): The number of samples for which $y_i = 0$ and $h_\partial(x_i) = 0$
- *False Positives* (*FP*): The number of samples for which $y_i = 1$ and $h_\partial(x_i) = 0$
- *False Negatives* (*FN*): The number of samples for which $y_i = 0$ and $h_\partial(x_i) = 1$

Some common performance measures for the supervised learning tasks will be examined next.

Accuracy (Equation 21) is the percentage of correct predictions by the model. It provides an assessment of the general performance of the model and the same formula can be generalized for multi-class classification tasks with the numerator being the sum of the samples classified

correctly. In multi-label classification tasks there exists an accuracy variant know as *Subset Accuracy*, which is the percentage of samples that had all their labels predicted correctly. A disadvantage of this metric is that it might be misleading in imbalanced datasets. For example, if 90% of the samples in a dataset have labels $y = 0$, then a models that predicts always $h_\vartheta(x) = 0$ will have Accuracy = 90%.

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN} \times 100\% \tag{21}$$

Recall (Equation 22) is the percentage of accurate positive predictions to the total samples with ground-truth positive labels.

$$Recall = \frac{TP}{TP + \mathbf{FN}} \times 100\% \tag{22}$$

Precision (Equation 23) is the percentage of the model's accurate positive predictions to the total samples predicted as positive.

$$Precision = \frac{TP}{TP + \mathbf{FP}} \times 100\% \tag{23}$$

In a multi-class setting, recall and precision are computed individually for each class. If the recall is 100%, the model has predicted all samples belonging to this class correctly ($FN = 0$). This might be misleading, as the model might be too aggressive and assign many other classes incorrectly. However, if the cost of FP is small and cost of FN is high, this behavior may be preferable. For example, when designing smoke alarms, it is preferable to have increased sensitivity (in the literature, sensitivity is another term for recall), even if that increases the number of false alarms.

On the other hand, if the precision is 100%, when the model assigns this class to a sample, its always correct ($FP = 0$).This might be misleading, as the model might be very cautious with respect to this class and only assign only the obvious samples. In certain scenarios, this cautious approach may be preferable, particularly in applications such as diagnosis of medical conditions.

Taking the above into account, there is a clear need for a performance measure that combines recall and precision. *F1-score* (Equation 24) is a function that meets this requirement and is widely used. This metric is the harmonic mean of recall and precision and is equal to 100% when both metrics are 100% and 0% if either of them is 0%.

$$F_1 = 2 \cdot \frac{Precision \cdot Recall}{Precision + Recall} \tag{24}$$

In multi-class and multi-label settings, the above per-class metrics are averaged to produce a total performance measure for the model. The averaging can be performed in many ways, the most widely used are *macro, micro* and *weighted* averaging [82, 97]. *Macro-averaging* computes the per-class metrics and then calculates their unweighted mean. This handles each class equally, regardless of their samples. *Weighted-averaging* uses the number of samples of each class as weights to calculate the weighted mean of the per-class metrics. Finally, *micro-averaging* computes the global *TP, FP, FN* and combines them into a global metric using the previously described equations.

Loss functions. Loss or cost functions are a subcategory of performance measure functions that assess how far the model's predictions are to the ground truth labels and are usually differentiable with respect to the parameters ϑ. In simple linear models, they allow for estimation of

the parameters using analytical differentiation and closed form solutions. In more complex non-linear models they allow for the application optimization methods like the iterative algorithms in the *Gradient Descent* family. Some common loss functions are presented next.

Cross Entropy Loss (CEL) (Equation 25) is a loss function mostly used in classification tasks and can be theoretically derived from the MLE principle. Suppose that $\hat{h}_\vartheta$ is a the model function that assigns class probabilities to the samples of the dataset. Then CEL measures the difference between a predicted class probabilities, $\hat{h}_\vartheta(x_i)$, and the ground truth labels.

$$CEL = -\frac{1}{N}\sum_{i=1}^{N} y_i \log(\hat{h}_\vartheta(x_i)) \tag{25}$$

Mean Squared Error (MSE) (Equation 26) is a loss function mostly used in regression tasks that measures the squared difference between the predicted values and ground truth values. The squaring of the error penalises larger errors more and MSE is often referred to as the L_2 loss function because it uses the L_2 norm of the error vector. Interestingly, in some cases, MSE can be used for MLE when we assume that the dataset is i.i.d. and the distribution $p(y|x)$ is Gaussian distribution[32].

$$MSE = \frac{1}{N}\sum_{i=1}^{N}(y_i - h_\vartheta(x_i))^2 \tag{26}$$

Mean Absolute Error (MAE) (Equation 27) is a loss function mostly used in regression tasks that measures the absolute difference between the predicted values and ground truth values. MAE linearly handles all errors and therefore larger errors are not disproportionately more costly. As a result, training with MAE is less sensitive to outliers and dataset noise. MAE is often referred to as the L_1 loss function because it uses the L_1 norm of the error vector.

$$MAE = \frac{1}{N}\sum_{i=1}^{N}|y_i - h_\vartheta(x_i)| \tag{27}$$

Smooth L_1 Loss (SL_1) is a function that combines the MSE and MAE using the parameter β. It produces squared error if the absolute element-wise error falls below beta and an absolute error otherwise. Its outlier sensitivity falls between the MSE and MAE losses.

$$\begin{aligned} l_i &= \begin{cases} 0.5\frac{(y_i - h_\vartheta(x_i))^2}{\beta}, & \text{if } |y_i - h_\vartheta(x_i)| < \beta \\ |y_i - h_\vartheta(x_i)| - 0.5\beta, & \text{otherwise} \end{cases} \\ SL_1 &= \frac{1}{N}\sum_{i=1}^{N} l_i \end{aligned} \tag{28}$$

Regularization. As seen above loss functions main goal is to guide the learning process to statistically estimate the parameters to produce predictions that are close to the ground truth labels. To minimize the capacity of the model and thus avoid overfitting, some penalty terms can be added to the loss function. This technique is known as regularization. The penalty terms create a competitive optimization problem that prevents the model from fully optimizing the objective of the original cost function [32]. Two common regularization methods are L1 and L2 (Equations 29 and 30).

$$L_1\text{regularization term} : \lambda\sum_{i=1}^{m}|\vartheta_i| \tag{29}$$

$$L_2\text{regularization term} : \lambda \sum_{i=1}^{m} \theta_i^2 \tag{30}$$

With λ being the parameter that controls regularization strength and θ_i the model parameters.

Logistic Regression. Goodfellow et al. [32] make the observation that in most cases machine learning frameworks require four components: a dataset, a loss function, an optimization algorithm and a model. Choosing the instantiation of each component requires careful consideration of the task at hand and experimentation. In this section, we will focus on Logistic Regression, a fundamental ML algorithm. This algorithm is within the scope of this thesis because it is used as a evaluation approach for the proposed representation learning methods.

In order to provide context for the experiments that follow, suppose that we have two encoder models e_1 and e_2, and we want to compare their performance on a classification dataset $\mathcal{X} = \{(x_1, y_1), \ldots, (x_N, y_N)\}$. The output vectors of the encoders $\mathbf{z}_i = e(\mathbf{x}_i)$ are of size D_z. The Logistic Regression model will take the weighted sum of the representation vectors $\mathbf{z}_i$ as described in Equation 31. The weights $\mathbf{w}$ and the bias term b are the parameters of the model. t_i is called the *logit* and is transformed into the probability that this sample belongs to the class using the *Sigmoid* function (Equation 32). In order to make the final classification, a threshold must be set, usually at 0.5 over which the sample is classified as belonging to the class [18].

$$t_i = \left(\sum_{j=1}^{D_z} w_j z_{i,j} \right) + b \tag{31}$$

$$p(y|x_i) = \sigma(t_i) = \frac{1}{1 + e^{-t_i}} \tag{32}$$

In the multi-class variant of Logistic Regression, t_i is a vector of logits, one for every class. The logits are transformed into class probabilities using the *Softmax* function (Equation 33).

$$p(y_c|x_i) = \text{softmax}(t_i) = \frac{e^{t_{i,c}}}{\sum_{j=1}^{K} e^{t_{i,c}}} \tag{33}$$

During model training, the cross-entropy loss is minimized by estimating the parameters using Maximum Likelihood Estimation (MLE). As logistic regression is a generalized linear model (with logits originating from a linear function), the loss function is convex, ensuring the presence of at most one global minimum. The optimal parameters that achieve this global minimum can be found using optimization algorithms. Gradient descent iteratively changes the weights in the direction that minimizes the loss function using the gradient of the loss with respect to the weights. Alternatively, the Limited-memory Broyden-Fletcher-Goldfarb-Shanno (L-BFGS) algorithm [11] approximates the inverse Hessian matrix to achieve faster convergence, especially in high-dimensional problems.

After training and evaluating the performance of the two classifiers, we can draw some conclusions about the representation vectors produced by the two encoders. Methods that use linear models (such as classifiers or regressors) on top of frozen encoders to evaluate their representations are termed linear probing [86, 45].

3.2 Deep Learning

Deep learning is a branch of ML that has achieved many successes in recent years in fields such as computer vision and natural language processing, and tasks such as image and speech recognition, autonomous driving, and healthcare. In the deep learning framework, models can take various forms, including feedforward neural networks (Figure 14), recurrent neural networks (RNNs) for sequential data, and convolutional neural networks (CNNs) for image processing. Each architecture is designed to approximate functions in different domains and introduce different inductive biases. Their main advantage of deep learning techniques compared to other ML methods is their ability to automatically learn representations from raw data, eliminating the requirement for hand-crafted representations [62].

Neural networks are made up of multiple layers. Each layer uses non-linear transformations to process its input and produces an output that is transmitted to the next layer. The first and last layers are referred to as *input* and *output* layers respectively and the intermediate layers are called the *hidden* layers. The *depth* of the neural network is the number of layers it contains and its *width* is the size of its hidden layers. Increasing the width and depth of a neural network will generally enhance its capacity and ability to approximate more complex functions. However, this also requires more data and computational resources to optimize the parameters.

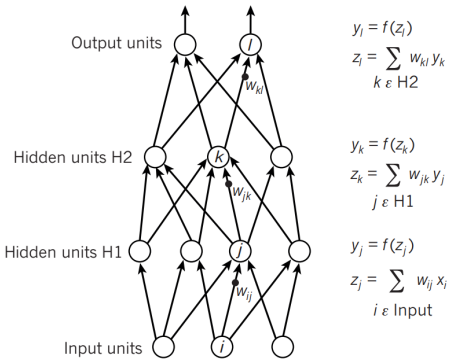


Figure 14: The architecture of a feedforward neural network with two hidden and the equations of the *forward pass* that describe how the values are computed at each layer to produce the model's prediction. Source: [62]

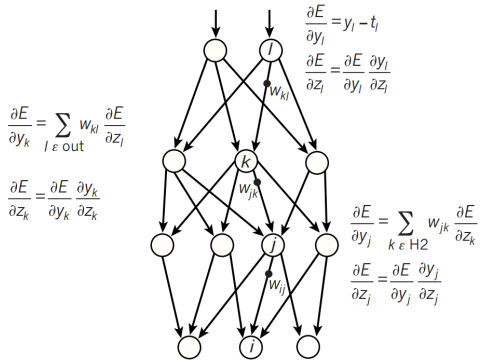


Figure 15: The backward differentiation flow and equations of the *backward pass* of the *backpropagation* algorithm. Source: [62]

As shown in Figure 14, the layers of a feedforward neural network are made up of neurons. Each neuron in a layer is connected to neurons in the next layer through weighted connections w_{ij}. The input to each neuron is a weighted sum of the outputs of the previous layer H, represented by $z_i = \sum_{j \in H} w_{ji} y_j + b$ (the bias term b is omitted from the figure for simplicity). This sum is then passed through an non-linear *activation function* $f(z_i)$ to produce the neuron's output. This non-linearity allows the neural network to learn complex patterns [32, 62]. Some common activation functions are the following.

- The *Sigmoid*: function maps inputs to values between 0 and 1,

$$f(z) = \frac{1}{1 + e^{-z}} \tag{34}$$

- The *Hyperbolic Tangent (tanh)* that maps inputs to values between -1 and 1.

$$f(z) = \frac{e^z - e^{-z}}{e^z + e^{-z}} \tag{35}$$

- The *Rectified Linear Unit (ReLU)* is a computationally simple and widely used activation function.

$$f(z) = \max(0, z) \tag{36}$$

In deep learning, optimizers are mainly iterative gradient-based algorithms. This is because the non-linear nature of neural networks results in non-convex loss functions. As a result, there is no guarantee that the training process will converge to a global minimum. One of the most common loss functions in deep learning is the cross-entropy loss, with the output of the output layer being conditional probabilities $p(y|x)$ with the use of sigmoid or softmax activation functions.

Stochastic Gradient Descent. Stochastic Gradient Descent (Algorithm 1) and its variants are the most widely used optimization algorithms in deep learning. The algorithms iteratively update the weights of the model using gradients, $\hat{g}$, computed on small random subsets of data called mini-batches. This estimates the gradient that would result from the whole dataset and as a result can display some variability, making the optimization process noisy. This provides SGD with its stochastic nature that can help avoid convergence to local minima. The gradient $\hat{g}$, being vector that points in the direction of the most rapid ascent of the loss function, is used to update the network's parameters with the rule presented in line 5 of Algorithm 1, where ϵ_k is the learning rate at iteration k. This learning rate may vary according to a schedule to ensure stable convergence. Various extensions of SGD, such as momentum SGD, AdaGrad, and Adam, have been developed to address issues like slow convergence and oscillations near local optima [32].

Algorithm 1 Stochastic gradient descent (SGD). Adjusted from: [32]

Require: Learning rate schedule $\epsilon_1, \epsilon_2, \ldots$
Require: Initial parameters θ
1: $k \leftarrow 1$
2: **while** stopping criterion not met **do**
3: Sample a mini-batch of m examples, $B = \{(x_1, y_1), \ldots, (x_m, y_m)\}$ from the training set $\mathcal{X}_{train}$
4: Compute gradient estimate: $\hat{g} \leftarrow \frac{1}{m} \nabla_\theta \sum_i L(f(x^{(i)}; \theta), y^{(i)})$
5: Apply update: $\theta \leftarrow \theta - \epsilon_k \hat{g}$
6: $k \leftarrow k + 1$
7: **end while**

Backpropagation (Figure 15) is one of the most important building blocks of neural network training as it allows for an efficient computation of the gradient, $\hat{g}$, of the loss function with respect to each weight in the network. The algorithm consists of two stages: the *forward pass* and the *backward pass*. In the forward pass, input data is passed through the network to produce the model's predictions. During this pass the *computation graph* of the network is used to track the flow of data and compute intermediate values for each operation. The loss is then computed with the use of the ground truth labels and the loss functions. In the backward pass, the algorithm calculates the gradient of the loss with respect to each weight using the *chain rule of differentiation*. The gradients are *propagated backwards* through the network's computation graph, quantifying how each parameter contributes to the final loss [18, 32, 62].

Regularization for Deep Learning. As discussed in the previous chapter, *regularization* techniques aim to improve the model's generalizability by introducing some constrains or penalties in the learning process. Regularization is important for neural networks to prevent overfitting caused by their flexibility and their ability to model complex functions. Some of the most important regularization techniques for deep learning are presented next.

- *Parameter Norm Penalties* include penalty terms introduced to the loss function, like the L_1 and L_2 regularizers that were previously discussed. Both methods introduce a preference for simpler models. L_2 regularization works well for most models and L_1 regularization is useful in cases that require sparse models, as the optimization objective it introduces tends to push some parameters to zero.

- *Dataset Augmentation* includes methods that increase the size of the training dataset by modifying the existing data. Methods that fall into this category have been proven particularly successful in image classification, object detection and segmentation. Common image augmentations include random rotations, flips, translations, scaling, adding noise to the original images or combining images from different categories. For the tasks of object detection and segmentation some techniques combine different backgrounds with objects. Building upon these concepts, certain techniques use *generative models* to produce *synthetic data* that mimic the original dataset's statistical properties.

- *Multitask Learning* involves training a model on multiple tasks simultaneously. As a result the model learns representations that are useful for all the tasks, which makes it less likely to overfit to any one particular task.

- *Early Stopping* is a method that utilizes the validation loss to stop the training before overfitting occurs. In general, training and validation loss decrease during training until a certain point at which the validation loss begins to increase again. This is a strong indication that the model has started to overfit by learning the noise and spurious correlations of the training set and the model's ability to generalize will not improve further.

- *Parameter Sharing* is a form of regularization that forces a set of neurons to share weights. This reduces the amount of learnable parameters and is a way to introduce a preference for simpler models and domain knowledge into the frameworks. In computer vision, Convolutional Neural Networks (CNNs) are an architecture that successfully exemplifies this method by applying the same convolutions filters across an image, taking advantage of the translation invariance of this domain.

- *Dropout* [101] is a simple but effective regularization technique. During training, at each iteration, each neuron and its connections are retained with probability p. During testing, all the neurons are on, with their weights scaled by p. This allows the network produce the same expected output in both cases and reduce overfitting by preventing its heavy reliance on a small set of neurons.

Convolutional Neural Networks. Convolutional Neural Networks (CNNs) were one of the first deep learning architectures to demonstrate the potential of deep learning. In the field of computer vision, CNNs have long been the predominant neural network architecture, accomplishing numerous breakthroughs and pushing the boundaries of what was previously thought possible in the field. Their ability to automatically learn hierarchical features from raw image data has transformed tasks such as image classification, object detection, and segmentation.

As discussed previously, this architecture utilizes the parameter sharing technique that makes the model and its representations equivariant to translation. This allows CNNs to recognize patterns regardless of their position in the image, as shifting the input equivalently shifts the output. This is a characteristic successful example of the use of inductive bias to improve computational efficiency [32].

However, in recent years, *self-attention* based architectures such as the *transformer* have been gaining popularity in computer vision, showcasing that large-scale training with minimal inductive bias outperforms inductive bias [21]. This aligns with Sutton's observation in AI research [103], which highlights that models leveraging scalable computation tend to eventually outperform those relying heavily on human-engineered features or domain-specific knowledge. In recent years, hybrid CNN-Transformer methods have emerged, such as [126, 66], especially when large datasets are unavailable. In this thesis, the primary emphasis is placed on models based on the transformer architecture. As a result, the next section will introduce the main transformer tool for computer vision, the *Vision Transformer*.

Vision Transformer. The *Transformer* [111] is a neural network architecture that was originally proposed for the task of machine translation. Recurrent neural networks (RNNs), such as long short-term memory (LSTM), were previously considered the state-of-the-art in processing sequential data for natural language tasks. The transformer architecture's main difference to these models is its ability to process the sequences in parallel. This allows transformers to be trained on more data in less time, scale to much larger sizes and better exploit long-range dependencies in the data.

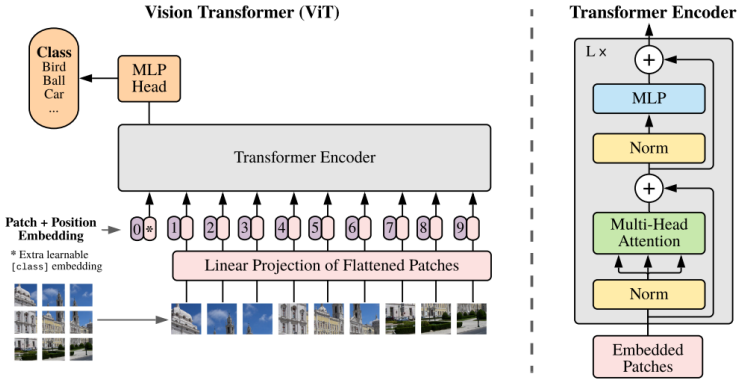


Figure 16: The *Vision Transformer* and *Transformer Encoder* architectures. Source: [21]

The *Vision Transformer* (ViT) applies the transformer encoder architecture to image inputs [2]. The key idea is to split an image into patches of fixed size. Each patch is a token, and the entire set of tokens is processed using *self-attention*, like words in a sentence.

More formally, given an input image $x \in \mathbb{R}^{H\times W\times C}$, where H and W are the height and width of the image, and C is the number of color channels, the image is divided into $N = HW/P^2$ patches, x_p of size P^2C. Each patch, $x_p^{(i)}$, is transformed into a *patch embedding*, $z^{(i)}$, by being flattened and projected into a D dimensional space using a learnable linear projection, E. A vector of learnable parameters, termed positional embedding, E_{pos}, is added to the transformed patches

[2] ViTs for video inputs are discussed in subsequent sections of the thesis.

to provide information about the spatial structure of the image. Finally, [class], a special D dimensional vector of learnable parameters, is prepared as part of the input to model's the next processing stage. [class] serves as the initialization of a vector that at the end of the model's computation provides a representation for the whole image. It is beneficial to conceptualize all the patch embeddings as being combined into a matrix $\mathbf{z} \in \mathbb{R}^{N+1 \times D}$.

$$z^{(i)} = x_p E + E_{\text{pos}} \in \mathbb{R}^D, \quad \forall\, i \in N \tag{37}$$

Subsequently, L identical layers process the patch embeddings. Each layer consists of the following components:

- *Layer Normalization* [63]: $\mathbf{x} = LN(\mathbf{z})$
- *Multi-Head Attention*: $\texttt{MHA}(\mathbf{x})$
- Residual Connection: $\mathbf{x} = \mathbf{x} + \texttt{MHA}(\mathbf{x})$
- *Layer Normalization*: $\mathbf{x} = LN(\mathbf{x})$
- *Multi-Layer Perceptron*: $\texttt{MLP}(\mathbf{x}) = W_2 \cdot f(W_1 \cdot \mathbf{x} + b_1) + b_2$ where W_1 and W_2 are weight matrices, b_1 and b_2 are biases, and f is the activation function.
- *Residual Connection*: $\mathbf{x} = \mathbf{x} + \texttt{MLP}(\mathbf{x})$

In all steps $\mathbf{x} \in \mathbb{R}^{N+1 \times D}$.

Multihead self-attention, which is the core module of the transformer architecture. Given the patch embedding matrix $\mathbf{z}$, the self-attention mechanism at the head h computes the attention scores for each token using the query matrix $\mathbf{q}_h \in \mathbb{R}^{(N+1) \times d_k}$, the key matrix $\mathbf{k}_h \in \mathbb{R}^{(N+1) \times D_h}$ and the value matrix $\mathbf{v}_h \in \mathbb{R}^{(N+1) \times D_h}$.

$$\mathbf{q}_h = \mathbf{z} W_{q,h}, \quad \mathbf{k}_h = \mathbf{z} W_{k,h}, \quad \mathbf{v}_h = \mathbf{z} W_{v,h} \tag{38}$$

Where $W_{q,h}, W_{k,h}, W_{v,h} \in \mathbb{R}^{D \times D_h}$ are learned projection matrices, and D_h is the dimension of the key and query vectors which typically is $D_h = D/H$, in a multi-head attention module with H heads. The attention matrix for every head is computed in the following way:

$$\mathrm{A}_h(\mathbf{q}_h, \mathbf{k}_h, \mathbf{v}_h) = \mathrm{softmax}\left(\frac{\mathbf{q}_h {\mathbf{k}_h}^T}{\sqrt{D_h}}\right)\mathbf{v}_h \tag{39}$$

$$\mathrm{SA}_h = \mathrm{A}_h \mathbf{v}_h \tag{40}$$

The attention weights $A_{i,j}$ symbolize the importance that token i allocates to the patch embedding of token j when computing its representation. In multi-head attention, each head is set to capture different relationships in the input data. The outputs from each attention head are concatenated and projected back into the D-dimensional space.

$$\mathrm{MHA} = [\mathrm{SA}_1; \ldots; \mathrm{SA}_H] W_O \tag{41}$$

Where $W_O \in \mathbb{R}^{(H \times D_h) \times D}$ is a learned projection matrix.

The output of the ViT model consists of $N + 1$ context-aware representation vectors, $\mathbf{x}_i$. Here, $\mathbf{x}_0$ corresponds to the representation vector of the [class] token, which can be used as a representation of the entire image. Transformers can be trained using labeled datasets in a classification setting but are also in self-supervised representation learning settings. Most of the models relevant to this thesis employ the latter approach and, most specifically, the method of *Auto-encoding* where the transformer encoder is followed by a decoder module that attempts to reconstruct the input. The next section provides an analysis of the auto-encoder framework and other representation learning techniques.

3.3 Representation Learning

Representation learning is the field that aims to develop methods by which models automatically extract useful representations from raw input data. These learned representations are then used for downstream tasks. Outside of ML, AI methods are based on hand-crafted features and algorithms, which is known as *feature engineering*. In traditional computer vision, this includes methods such as edge detection, color histograms, or descriptors such as SIFT (Scale-Invariant Feature Transform) [68] and HOG (Histogram of Oriented Gradients) [16]. These methods relied on domain expertise and the process was often time-consuming and inflexible [74].

In contrast, representation learning in ML, especially after the latest advances in neural network methods, automates this process by allowing models to learn representations directly from the data. This end-to-end learning process reduces the need for human intervention and often leads to better generalization and performance. In deep learning, representation learning methods often utilize self-supervised learning, training encoders in *pretext tasks* that do not require labeled datasets.

The success of deep learning is often attributed to the ability of neural networks to learn the non-linear manifolds in the data distributions. These advancements have given rise to the *manifold hypothesis*, which suggests that high-dimensional data, such as images, audio, or text, are concentrated in regions with fewer dimensions. In the ML literature, the term *manifold* [3] loosely refers to a lower-dimensional space within a higher-dimensional space, where a connected set of points can be approximated using fewer degrees of freedom than the higher-dimensional space. [4, 32]

Auto-encoders. One deep learning method that takes advantage of the ideas outlined by the manifold hypothesis is the *auto-encoder* (AE). The AE framework falls into the self-supervised learning paradigm as it attempts to reconstruct the input. In general, representation learning aims to train an encoder $e : \mathcal{D} \to \mathcal{Z}$ that maps raw input data, $x \in \mathcal{D}$, to representation vectors, $z \in \mathcal{Z}$. In AEs, the reconstruction of the input is created by a module called the *decoder*, that can be formalized as a function $g : \mathcal{Z} \to \mathcal{D}$. A common loss function for training AE is the *reconstruction loss*, typically defined as the difference between the input and the reconstructed output (Equation 42).

$$L_r = \frac{1}{N} \sum_{i=1}^{N} ||x_i - g(e(x_i))||, \tag{42}$$

The most widely used AEs are those termed *undercomplete* (Figure **??**), which attempt to reconstruct the input after mapping it to a representation space of significantly smaller dimensionality. To push models to extract useful representations, various modifications of the undercomplete AE have been proposed. Two notable types are the *Denoising* AEs and *Masked* AEs [121].

Denoising Auto-encoders (DAEs) [32] learn from data that has been corrupted by noise. A noisy version of the input, $\tilde{x}$, is fed into the encoder, and the model is tasked with reconstructing the original input, x:

$$L_r = \frac{1}{N} \sum_{i=1}^{N} ||x_i - g(e(\tilde{x}_i))||, \tag{43}$$

Masked Auto-encoders (MAEs) [121] attempt to reconstruct an input, $\tilde{x}$, that has some parts hidden. In computer vision, MAEs are particularly effective both in image and video processing

[3] A more rigorous definition of the concept of a manifold can be found in the mathematical *Topology* literature.

and are some of the main modules of the models used in this thesis. More details and specific techniques are discussed in Chapter 4.

Object-centric representation learning. Object-centric representation learning is a developing paradigm in ML, where the goal is to represent complex environments in terms of objects, rather than treating the entire scene as a single entity and extracting a single representation. This paradigm is compatible with the proposed "principles of grouping" (Figure 17) from psychology and cognitive science, which account for how humans process visual signals by organizing them into objects [77].

Object-centric representation learning has shown the potential to improve the generalization capabilities and sample efficiency of models as well improve interpretability. However, in the past, these techniques were hindered by the difficulty and cost of dataset annotation when learning in a supervised manner. In recent years, this potential has been unlocked by architectures like Slot Attention that are self-supervised and highly scalable. As a result, they can use large unlabeled image and video datasets [67, 8, 3]. The Slot Attention architecture is a key focus of this thesis and is discussed further in chapter 5.

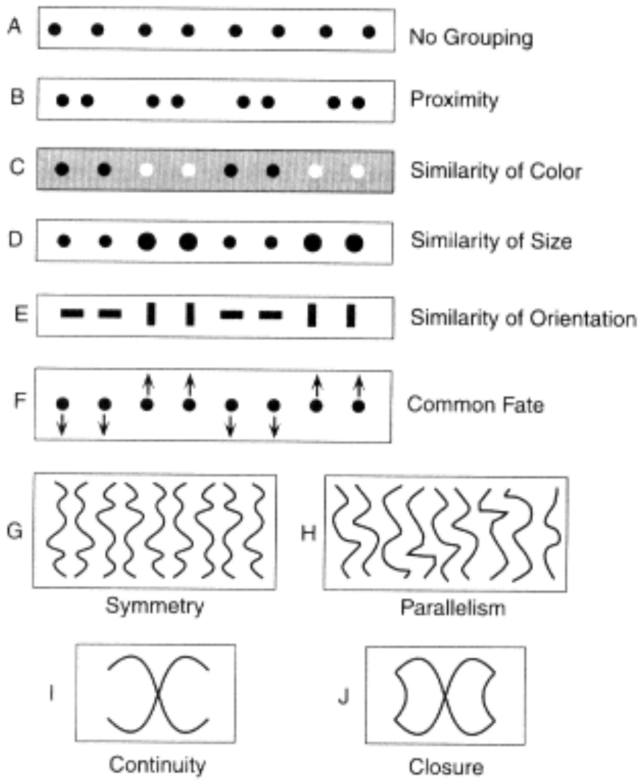


Figure 17: Principles of grouping (or Gestalt laws of grouping): some of the factors that govern which visual elements are perceived by humans as going together. Source: [77]

Human-aligned representations and saliency methods The growth and success of deep learning in recent decades have increased interest in explainability and its alignment with hu-

man preferences. Useful techniques interpret the model's decisions by highlighting which parts of the input were most influential [37, 47]. In robotics, being able to predict and align with where humans pay attention is critical to improving human-robot interaction and collaboration.

Attention and saliency are two terms that are useful in this context. In general, *attention* is a top-down process that incorporates expectations and preconceived knowledge to process the sensory signal by highlighting areas of importance. On the other hand, *saliency* is mainly a bottom-up process that uses low-level characteristics of the input to identify regions of potential importance [25, 131, 59, 58].

Deep learning supervised methods have proven to be highly successful in predicting human generated saliency maps using multi-modal data [60, 110, 20]. Recently, self-supervised approaches utilizing video data have demonstrated the ability to align with human saliency and produce human-aligned representations without requiring ground truth maps for training [80].

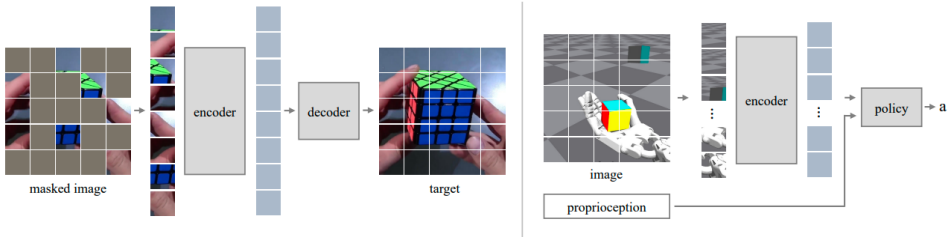


Figure 18: Image masked auto-encoder representation learning for robot control. Source: [87].

Representation learning for robot perception. In the field of robotics, recent years have witnessed an increased focus on data-driven approaches derived from ML for problems such as manipulation and planning, shifting away from analytical methods due to their generalization disadvantages [51]. The transfer of insights from the successful application of representation learning in natural language processing and computer vision plays a central role in these efforts.

Image representations play a crucial role in robotic manipulation tasks, where understanding of the environment is key to success. In particular, after the deep learning revolution, computer vision was equipped with effective representation extraction techniques that drove the paradigm of visuomotor policy learning (Figure 51) to produce many significant results [64, 106, 61, 95, 128]. The term *visuomotor* emphasizes that the flat state vector input to the policy model combines image representations with a vector that includes information about the robot's state, such as its pose or joint speeds.

Self-supervised learning techniques have gained traction in robotics representation learning, as they allow robots to leverage large amounts of unlabeled data. An indicative example is the success of Image Masked AEs in simulated[87] and real-world robotic tasks [88]. In the pre-training phase encoders were trained using images from egocentric datasets such as Ego4D [36] and EPIC-Kitchens [17] and *Hand-object Interaction* and action-centric datasets like Something-Something dataset [35]. The encoders are then frozen and used to learn control policies in various tasks (Figure 18).

4 Action to Object Knowledge Distillation

4.1 Introduction

Based on the positive results of visual pre-training methods that focus on modeling action-centric information, this thesis focuses on ways that modeling of datasets that capture the way humans interact with objects [36, 17, 35] can be incorporated in object-centric representations [67]. In this section we explore a method that aims to encode action experiences and associate them, through the use of video(of action)-to-image(of object) KD, with the depiction of the objects present in those experiences. In the following experiments the action experiences are in the form of videos of human actions from the Something Something v2 dataset [35], however similar techniques could be used to encode and create associations using the agents' own interactions with the objects.

These representations, can potentially be useful as a pre-training step in an object-centric modular learning framework such as that in Figure 19, where the individual object representations are becoming context-aware through a transformer module.

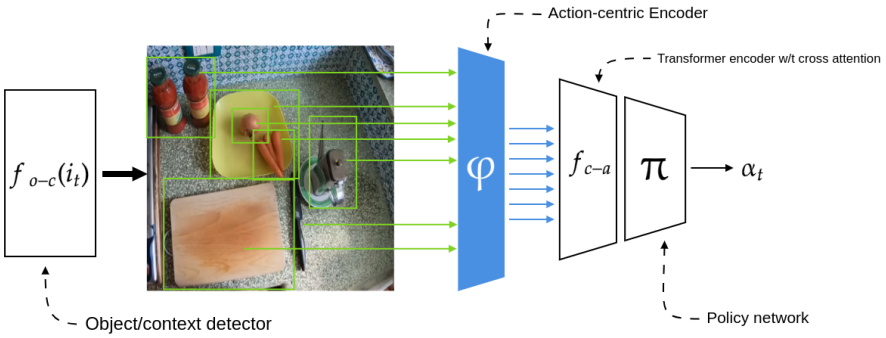


Figure 19: Example of AcE as part of a robotics modular learning framework.

Another potential usefulness of action-centric encoding could be to provide similarity metrics in an example retrieval setting, such as in the cloud database for advanced manipulation intelligence (Figure 20) proposed in [117]. The aim of such a database is to provide examples and information about certain tasks, with the examples being performed by humans or robots. When a robot is assigned a new task, it can retrieve examples from the database and also post information after completing a task. Object, representations can be used to retrieve useful examples. A similar idea could be applied to Augmented or Virtual Reality applications, where virtual assistants could provide affordance information and support, and might need to retrieve and provide action demonstrations [83].

The action-to-object distillation process transfers the knowledge of Video MAE to an image encoder, named the Object Action-centric Encoder (OAcE). The objective of OAcE is to model the representation space of videos featuring human actions on objects, originally accessible only by the Video MAE model. Using cross-modal distillation, we aim to make these representations accessible through a different modality, static images of objects (object crops).

The OAcE consists of two main modules:

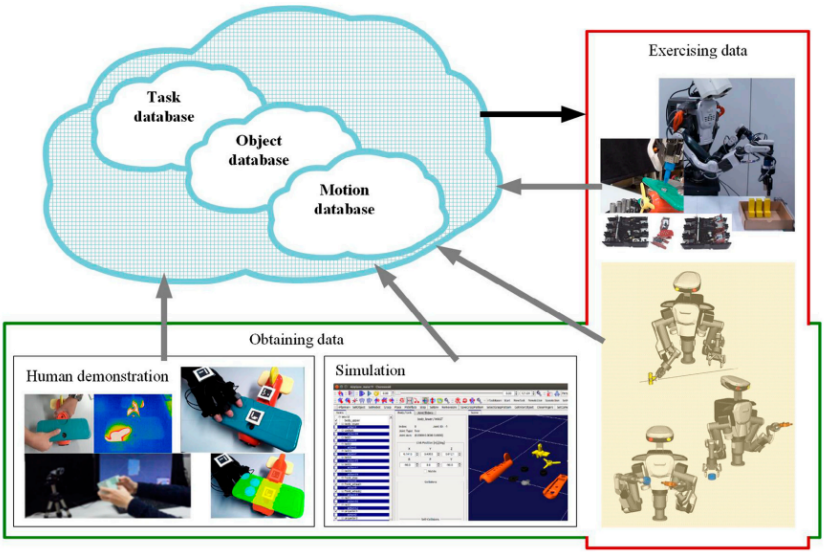


Figure 20: Cloud database for advanced manipulation intelligence. Source: [117]

- An **Image Encoder**, that transforms an image input from pixel space, to a dense image representation space, $I \in \mathcal{Z}_{img}$. In the following experiments, two different pre-trained models are used as Image Encoders, CLIP [86] and Image MAE [45].
- A **Mapping Module** that maps from the image representation space to the action-centric video representation space, $R \in \mathcal{Z}_{ac}$.

The primary goal of this experimental section is to explore if the OAcE can be compared with, and potentially enhance some of the state of the art static image encoders. The different methods are evaluated based on their performance in the task of affordance categorization. Before discussing the experimental methodology and results, the following section presents the theoretical background that inspired this study, along with the pre-trained models used as modules in the proposed framework.

4.2 Theoretical Background

4.2.1 Knowledge Distillation

Knowledge distillation (KD) [34, 53, 90] is a method of neural network compression in which a student model is trained to replicate the performance of a larger and more complex teacher model (Figure 21). This method was introduced, along with other model-reduction techniques like Network Pruning [14, 91, 31] to meet the need for models that are as effective as large deep models but run on devices with limited computational resources, such as mobile phones or autonomous cars. The effectiveness of the student model arises from the fact that it not only learns from the dataset, but also captures the ways that the teacher model generalizes. The implicit knowledge transferred is usually referred to as *dark knowledge*. This knowledge is usually acquired as a side effect of the training process and is not immediately obvious in the standard evaluation metrics, such as accuracy and recall. It can be observed in the way the model assigns (log) probabilities to the classes and includes nuance patterns in the training data

that help the model's decision making and generalization abilities, which cannot be acquired by simply training the smaller model on the dataset directly [48].

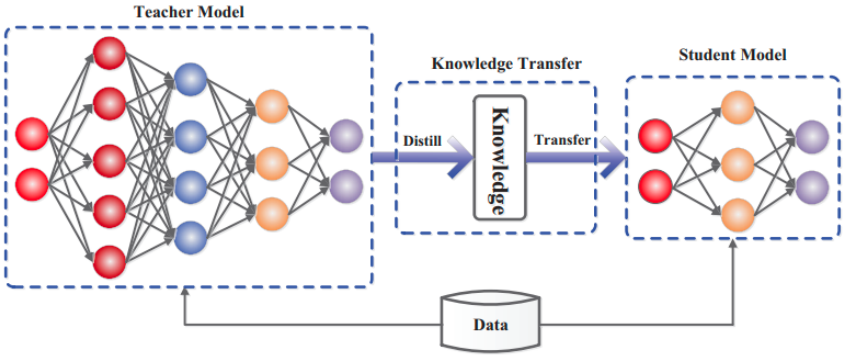


Figure 21: The generic teacher-student framework for knowledge distillation. Source: [34]

In recent years, several variations of KD were proposed, and extensive studies have explored the strengths and limitations of these frameworks [34, 53]. The KD categories that are relevant to the OAcE experiments are the following:

- **Feature-Based KD**: In this category of KD algorithms, the transferred knowledge is at a higher level compared to the Response-Based knowledge methods, where the student model targets the class probabilities of the teacher model. Feature-Based KD has shown promising results as a representation learning method [112, 26, 28]. Given its potential to capture and transfer the intricacies and transformation invariant aspects of the object and action recognition tasks as well as the action-object dynamics, it was considered a promising method to explore in this thesis.

- **Offline Distillation**: This means that the teacher is pre-trained prior to the distillation. This is the case with the Object Action-centric Encoder training procedure being attempted in this thesis, as the teacher is a pre-trained VideoMAE. However, since the training of the VideoMAE is self-supervised, potentially the same system could be applied in an online distillation manner, where the teacher and student learn simultaneously.

- **Cross-Modal distillation**: This means that the teacher's input is of a different modality than the student's. In our framework we attempt cross-modal distillation as the teacher encodes videos and the student attempts to distill the action-centric information into images of the objects. This falls in the category of video to image distillation or knowledge transfer [92, 80, 65].

- **Relational Knowledge Distillation**: This KD variant focuses on transferring the mutual relationships between data examples, as these relationships emerge in the representation space of the teacher model. The relationship of the samples is usually quantified through two types of loss function [79]: distance-wise loss and angle-wise loss. In distance-based loss, the Euclidean distances between pairs of samples are calculated in the output space, promoting the student to preserve distance relationships similar to those of the teacher. In angle-based loss, the angles formed by triplets of examples are considered, allowing a more detailed transfer of relational information.

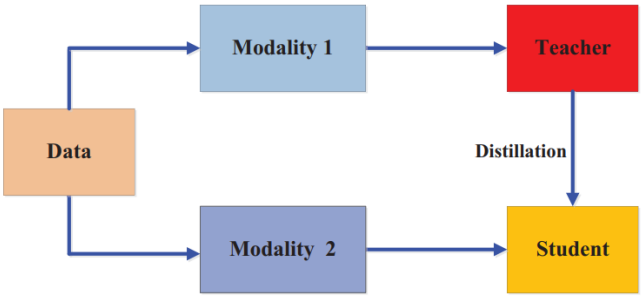


Figure 22: Cross-Modal Distillation. [34]

4.2.2 Image Encoders

The architectures and training methods of the image encoders used in these experiments are presented in the following sections.

CLIP [86]
The CLIP models presented by Radford et al. in "Learning transferable visual models from natural language supervision" [86] and this section presents some key elements from this study. CLIP models are among the most successful image encoders, in terms of generalization, flexibility, and efficiency. They are trained on a dataset that consists of captioned images and use the natural language's expressive ability to enrich a contrastive learning framework. The CLIP model is a pair of Transformer encoders $f_{\text{img}}, f_{\text{txt}}$ (Figure 23), one for each modality. Both inputs are first encoded in separate representation vectors $\mathbf{z}_{\text{img}} \in \mathbb{R}^{d_{\text{img}}}$, $\mathbf{z}_{txt} \in \mathbb{R}^{d_{img}}$ and then transformed to the multi-modal representation vectors, $\mathbf{I}, \mathbf{T} \in \mathbb{R}^{d_{mm}}$, using the learnable transformation matrices $\mathbf{W}_{\text{img}} \in \mathbb{R}^{d_{\text{mm}} \times d_{\text{img}}}$, $\mathbf{W}_{\text{txt}} \in \mathbb{R}^{d_{\text{mm}} \times d_{\text{txt}}}$ and L_2 Normalization as described in Equations 44.

$$\begin{aligned} \mathbf{I} &= L_2 Normalize(\mathbf{W}_{\text{img}} \cdot f_{\text{img}}(\mathbf{i})) \\ \mathbf{T} &= L_2 Normalize(\mathbf{W}_{\text{txt}} \cdot f_{\text{txt}}(\mathbf{t})) \end{aligned} \tag{44}$$

During training, for each batch, all images are combined with all text captions, generating $N \times N$ data points, of which N are positive $N^2 - N$ negative examples. The main goal of the training process is to bring the correct pairs closer while simultaneously pushing the incorrect pairs apart. This is achieved by computing the dot product [4] of all possible pairs of images and text, and minimizing a symmetric contrastive loss [124] between the similarities and the labels. The symmetry between the two modalities is achieved by computing two losses, the image-to-text contrastive loss (Equation 45) and the text-to-image contrastive loss (Equation 46), both using the Softmax Cross-Entropy Loss [85].

$$\ell_{\text{img}} = \sum_{i=1}^{N} -\log \frac{\exp(\mathbf{I}_i \cdot \mathbf{T}_i)}{\sum_{j=1}^{N} \exp(\mathbf{I}_i \cdot \mathbf{T}_j)} \tag{45}$$

[4] This computes the cosine similarity because the multi-modal representations are unit vectors due to the L_2 normalization in Equations 44.

$$\ell_{\text{txt}} = \sum_{i=1}^{N} -\log \frac{\exp(\mathbf{I}_i \cdot \mathbf{T}_i)}{\sum_{j=1}^{N} \exp(\mathbf{I}_j \cdot \mathbf{T}_i)} \tag{46}$$

As depicted in Figure 23 the positive examples are in the diagonal of similarity matrix, where $i = j$. The loss minimized during training is the mean of these two losses:

$$\mathcal{L}_{\text{CLIP}} = (\ell_{\text{img}} + \ell_{\text{txt}})/2 \tag{47}$$

A pre-trained CLIP model was chosen as one of the Image Encoder modules in OAcE due to its proven efficiency and task-agnostic characteristics. Another important takeaway from this paper is the detailed discussion on the evaluation of the representations. The authors make a strong case for zero-shot classification and linear probing (the fitting of a linear classifier on top of the representations) as evaluation metrics on the robustness and generalizability of the representations. The main advantage of these methods is that they are less likely to exploit spurious correlations, compared to other methods used to test representations on downstream tasks like the end-to-end fine-tuning of models. Linear probing is used as one of the evaluation methods of the OAcE representations.

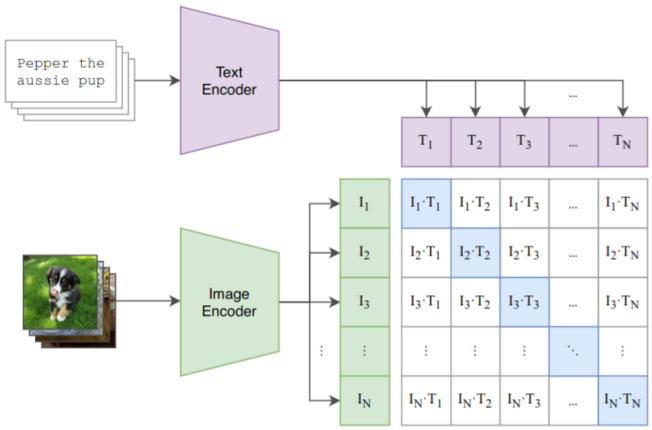


Figure 23: The CLIP contrastive framework. Source: [86]

Image Masked Auto-encoding [45]

The Image Masked Auto-encoder (MAE) that is used in the following experiments was proposed in the paper "Masked Autoencoders Are Scalable Vision Learners" [45] and this section presents some key elements from this study. Masked auto-encoding, is a self-supervised learning method that has been very successfully in the domain of natural language processing, shaping the training process of models like BERT [19]. The Image MAE aims to apply the method in the (static) image modality.

The key idea behind the MAE framework is that if a model can reconstruct a sample with some of its parts masked, then its encoder is likely to have learned high-quality representations. However, there are some important differences between language and vision that He et al.[45] address. Specifically, language, being a human-generated signal, is more information dense and as a result allows low masking ratios (15 %) to suffice in producing useful representations. In contrast, images have high information redundancy, requiring higher masking ratios (75

%) for the MAE to perform successfully. Furthermore, unlike BERT style models, where the decoder of the MAE is relatively simple, the Image MAE decoder is more complex and plays a more significant role in determining the quality of the learned representations, as it needs to reconstruct the input at pixel space, which is at a lower semantic level.

The Image MAE framework is based on the Encoder-Decoder Framework (Figure 24) with both modules adopting a ViT architecture. The encoder processes the tokens by linearly transforming them at first, then adding positional encoding, and finally passing them through several Transformer blocks. Unlike classical auto-encoder frameworks, the encoder only processes the unmasked tokens of the image input. This approach allows for efficient training by minimizing the data processed by the encoder.

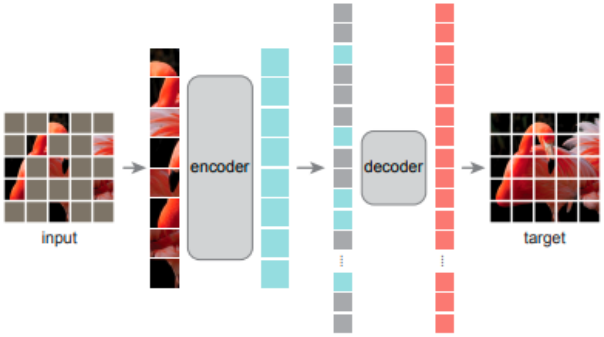


Figure 24: The image MAE framework. Source: [45]

The decoder, on the other hand, is designed to reconstruct the original image from both the encoded visible tokens and the masked tokens, which are added back after the encoding process. Each masked token is the same learnable vector that represents a patch that needs to be predicted. The decoder takes the encoded tokens, appends the mask tokens, applies positional encoding to all of them, and then processes this full set through its own series of Transformer blocks. The decoder is rarely needed after the self-supervised training process, and as a result its implemented as a smaller ViT model compared with the encoder. The encoder is the primary focus of the process and is designed to take up $> 90\%$ of the computation load per token.

The loss function in the Image Masked Auto-encoder (MAE) is the mean squared error (MSE) between the reconstructed and original pixel values[5], calculated only on masked patches. This approach is similar to the method used in BERT [19], where the loss is calculated only on the masked tokens. Focusing the loss calculation on the masked patches encourages the model to learn meaningful representations by reconstructing the missing parts, leading to better performance compared to using a loss function that includes all tokens.

The MAE framework has demonstrated strong transfer performance in downstream tasks such as object detection and segmentation. The encoder extracts image features either through the token class [21] or by averaging the token representations, with both approaches being equivalently effective.

[5]In recent years, some methods have proposed MAE frameworks that incorporate reconstruction of the input at feature level, through feature distillation. An example is MR-MAE (Mimic before Construct Masked Auto-encoder) [28], which adds a mimic loss that pushes the MAE to reconstruct the CLIP and DINO representations of the unmasked features. These methods not only improve encoder performance in downstream tasks, but also improve training speed by learning high-level and low-level semantics simultaneously.

4.2.3 Video Encoders

Video Masked Auto-encoding

The Video MAE model used in these experiments was introduced by Tong et al. at "VideoMAE: Masked Autoencoders are Data-Efficient Learners for Self-Supervised Video Pre-Training" [109] and this section presents some key elements from this study.. This paper presents a technique for extracting representations from videos using the self-supervised masked representation learning method. The authors address the challenges presented by the video modality, compared to the text and image modalities. The first challenge is the increased complexity introduced by the time dimension. The second challenge is that, in most cases, the useful signal is only a small percentage of the total input. Finally, when masking the tokens, the high temporal correlation between frames can lead to information leakage in parts of the video with little motion. To address these challenges, the authors recommend the use of a tubular masking technique (Figure 25) where the temporal neighbors of a token are also hidden. Additionally, to force the model to focus on the useful portion of signal and avoid spurious correlations they mask 90-95% of the total tokens.

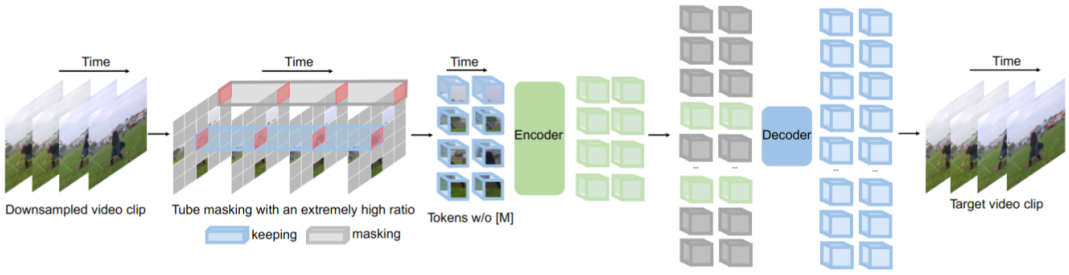


Figure 25: Video MAE. Source: [109]

The models trained with this method produce state-of-the-art results in the downstream task of action recognition. One of the datasets used for evaluating the Video MAEs is the Something-Something v.2 dataset. This dataset was chosen as the basis for the main part of the experimentation of this project, as it is lightweight and provides fine-grained action-centric information. A more in-depth discussion of the Something Something dataset can be found in corresponding section of this report. As a result, a Video MAE pre-trained on this dataset is used as a teacher model in the distillation process.

Masked Video Distillation [112]

In [112] Wang et al. introduce a multi-teacher KD technique to further improve the representations of the Video MAE. The authors propose a two-stage self-supervised representation extraction technique (Figure 26):

- **Stage 1:** Training two Masked Auto-Encoder *teacher models*
- **Stage 2:** Distill the representations of the *teacher models* to a *student model*

As *teacher models*, two different MAEs are used: one that produces image representations (MIM: Masked Image Modeling) and one that produces video representations (MVM: Masked Video Modeling). The paper presents experiments showing that the distillation of the MIM teacher improves the performance of the *student model* on problems where spatial information

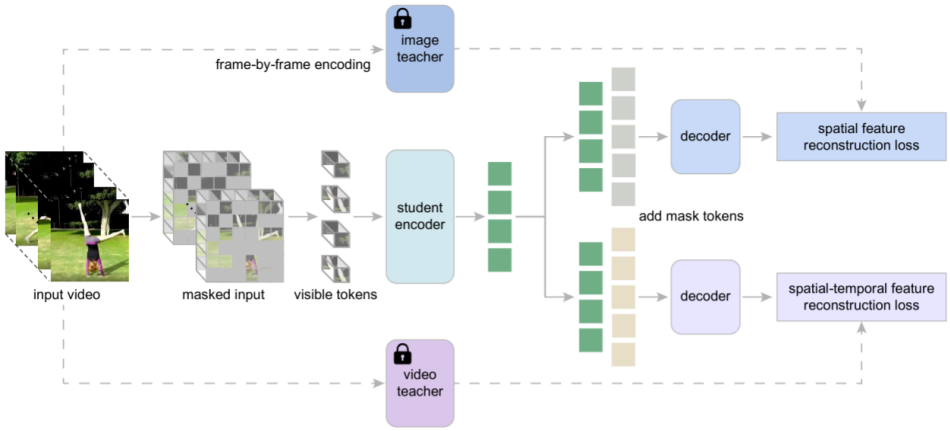


Figure 26: An overview of Masked Video Distillation framework. Source: [112]

is important (Kinetics 400) and the distillation of the MVM teacher improves the performance on problems where significant information is needed in the temporal dimension (Something-Something v2). The results are better when the KD process includes both teachers. This approach is reminiscent of results in the Action Recognition literature, inspired by biological studies, that demonstrate the benefits of a two-stream design: one stream focusing on spatial information and the other on temporal information. The first stream applies sparse temporal sampling with high resolution, while the second uses low resolution with denser sampling [27].

One of the main takeaways from this paper for this project was the effectiveness of feature-based knowledge transfer at the representation level. Additionally, a variant of the multi-teacher paradigm is used in the training of OAcE.

4.2.4 Affordance Categorization & Understanding

In the context of action-centric visual representations, the concept of affordances, which links object perception and action possibilities, provides a valuable perspective and inspiration for representation learning methods, as it occupies the space between what is objectively observable (object characteristics) and what is subjectively experienced (representations)[76, 13].

James J. Gibson suggested that for humans and animals, objects are not simply perceived as compositions of their qualities (shape, color, texture), but more crucially their affordances [30, 76]. The term affordance was coined by J.J.Gibson in 1977 and since then it has been used among researchers and students in many fields including psychology and neuroscience. According to Gibson [30]: "The affordances of the environment are what it offers the animal, what it provides or furnishes, either for good or ill. The word affordance implies the complementarity of the animal and the environment." Another important formalization of the term came from Sahin et al. [94], in which the authors aimed emphasise the fact that affordances can be viewed from different perspectives: the agent's, the observer's, and the environment's. According to them, all three must be taken into account when attempting to develop autonomous robotic agents.

The key terms relevant to this thesis are defined below.

- **Affordance Categorization**: This involves the multi-label classification of input images into a set of available affordances. This task usually acts as the foundation for more complex affordance recognition tasks.

- Affordance detection: The task of localizing and classifying objects based on the affordances. For n bounding boxes $X = \{x_1, x_2, ..., x_n\}$, the learning process should produce the function: $f : X \rightarrow Y$, where $Y = \{y_1, y_2, ..., y_n\}$ and $y_i = (r_i, l_i)$ with r_i representing the location of the bounding box and l_i the affordance label or a set of affordance labels.

Affordance categorization is a suitable candidate for evaluating object representations intended for robotics. This is due to the fact that identifying potential actions in an enviroment can help the agent plan and collaborate with humans or other robots [13, 42].

4.3 Datasets

4.3.1 Something-something v2

The Something-Something v.2 dataset [35] presents a fine-grained approach to the Action Recognition task. It consists of 220,847 videos that belong to 174 categories of actions. The dataset creation protocol allowed the people creating the videos to choose an action label and then perform it on an object of their choice, resulting in a diverse range of scenarios and action-object pairs. In the process of dividing the dataset into training and test sets, videos created by an individual are included either entirely in the training set or in the test set.

(a) A sample from the Something Something dataset with the action label: "Letting something roll along a flat surface".

(b) A sample from the Something Something dataset with the action label: "Squeezing something".

Figure 27: Samples from the Something-Something dataset [35]

The action categories are curated with the aim of pushing the models to deepen their understanding of the physical world and develop a form of "common sense". To perform well in this data set, the models need to distinguish between genuine and fake actions. For example, the models are required to learn to distinguish between "Putting something behind something" and "Pretending to put something behind something (but not actually leaving it there). Video representations that perform well in this dataset are likely to have captured a high-quality action-centric signal from the videos. The state-of-the-art performance of VideoMAE [109] and MVD [112] in this dataset is presented in Table 10. These models were considered to be good

candidates for the role of Teacher Models in the process of video-to-image knowledge distillation. From the models shown in Table 10, only the ViT-S and ViT-B versions of the VideoMAE framework were accessible as pre-trained models from the authors. Ultimately, the ViT-S variant was deemed sufficient for the proof-of-concept experiments of this thesis, as the performance improvement of the larger ViT was only marginal.

Method	Backbone	Top-1	Top-5
MVD	ViT-S	70.7	92.6
MVD	ViT-S	70.9	92.8
MVD	ViT-B	72.5	93.6
MVD	ViT-B	73.7	94.0
MVD	ViT-L	76.1	95.4
MVD	ViT-L	76.7	95.5
MVD	ViT-H	77.3	95.7
VideoMAE	ViT-S	66.8	90.3
VideoMAE	ViT-B	70.8	92.4
VideoMAE	ViT-L	74.3	94.6
VideoMAE	ViT-L	75.4	95.2

Table 10: Performance comparison of MVD and VideoMAE models with different backbones and configurations on the Something Something dataset. Sources: [109, 112]

The initial experiments were conducted using a small subset of the dataset, which was manually annotated using the Supervisely Computer Vision Platform [102]. During this phase, a small subset of videos was selected, and bounding boxes were annotated around the interacting objects. During the main phase of the experiments, the bounding boxes and annotations from the Something-Else [72] dataset were used, which is an extension of the Something-something dataset.

4.3.2 Something-Else

The Something-Else [72] dataset is an extension of the Something-Something dataset that introduces new annotations (Figure 28) and data splits. This extension provides bounding boxes of the hands and objects involved in the action, which was essential in assessing the OAcE model. The main goal of the dataset is to test models on the task of compositional action recognition, that enhances the action recognition task by focusing on compositional generalization. The compositional generalization ability of a model is its ability to adapt and recompose knowledge acquired in the past to novel and unfamiliar contexts [72, 114]. In the context of action recognition that means that the dataset challenges models to recognize actions with unseen combinations of verbs and nouns.

To achieve this, the category of frequent objects (the objects that appear more than 100 times in the dataset) is split into two disjoint groups A and B, and action categories are divided into groups 1 and 2. The training set combines action group 1 with object group A, and action group 2 with object group B (1A+2B), while the validation set flips this (1B+2A). For example, in the training set, the model might learn actions such as "Pick up Cup" and "Place Book" (Group 1 with Group A), as well as "Open Phone" and "Close Pen" (Group 2 with Group B), while the test set challenges the model with new combinations like "Pick up Phone" and "Place Pen" (Group 1 with Group B), along with "Open Cup" and "Close Book" (Group 2 with Group A). Drawing inspiration from this concept, one of the splits of the dataset presented in the following section attempts to

test the models generalization ability in predicting the affordance of unseen categories of objects.

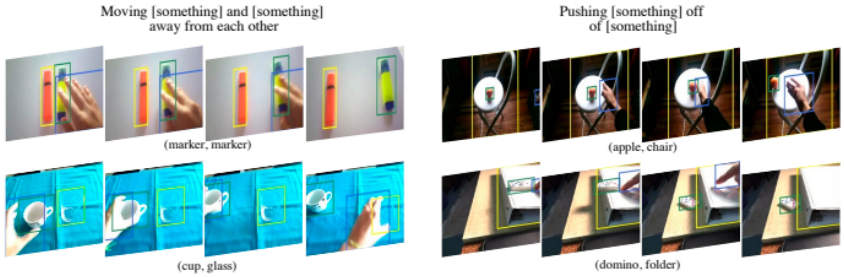


Figure 28: Something-Else annotations. Source: [72]

4.3.3 Something's Affordances: Curating a Small-Scale Affordance Categorization Dataset

Something's Affordances is a small-scale dataset that extends the Something-Else dataset and focuses on affordance categorization. To provide a proof of concept for the proposed methods, which are aimed at enhancing image representations through the distillation of knowledge present in videos of actions, a small subset of action categories was selected based on their ability to test the representations. For instance, the action 'Putting something on a surface' translates into affordances like 'graspable' or 'movable,' which are applicable to almost all objects in the dataset. Conversely, actions like 'rolling an object on a flat surface' are more useful for demonstrating whether the model has learned specific characteristics that make an object 'rollable.' The selected action categories and the corresponding affordances are presented in Table 11.

Affordance	Something-Something action labels	# video samples
Foldable	Folding something, Unfolding something	1620
Rollable	Rolling something on a flat surface, Letting something roll up a slanted surface, so it rolls back down, Letting something roll down a slanted surface, Letting something roll along a flat surface	2913
Squeezable	Squeezing something	2202
Containment	Pouring something out of something, Pouring something into something until it overflows, Pretending to pour something out of something, but something is empty, Showing that something is empty	2289
Tearable	Tearing something just a little bit	1620

Table 11: Something's Affordances labels and the corresponding action labels from the Something-Something dataset.

One of the limitations of extracting object crops from a video dataset is that many samples contain interference from hands or other objects To minimize this issue, object crops were extracted from the first 10 frames of the videos, where the objects typically appear on their own. Furthermore, due to camera or hand-object movement, some of the object crops only depict part of the object or exhibit motion blur (Figure). This is in contrast to other affordance categorization datasets, such as [55], which contain clear, unobstructed images of objects. Although this may initially seem like a drawback, it can actually be beneficial. These variations can simulate the

effects of image augmentation techniques, which are used to artificially make trained models more robust [116].

In line with the Something-Else dataset [72], we define the *frequent objects* subset, witch consists of the objects that appear more that 20 times in the videos. This is to ensure that the objects appear in enough examples that affordance information can be extracted from the dataset statistics. In total the dataset consists of 11,235 videos, out of which 123,434 object crops are extracted. For every object in *frequent objects* set we calculate the frequency distribution of the actions. From this frequency distribution, we extract the multi-label affordance targets for each object, by thresholding the frequencies in a way to avoid objects that are utilized in an uncommon manner (outlier scenarios). For example, the action frequency distribution and the multi-label affordance targets for the object "bottle" are presented in Table 12.

The dataset is split in two ways: **Video-based split** and **Object-based split**.

1. **Video-based split**: In this split, the dataset is divided into three sets (train, validation, test), ensuring that images from the same video belong to the same split.

2. **Object-based split**: This split targets (compositional) generalization by dividing the objects into two sets, *Set A* and *Set B*. *Set A* is used for training, while *Set B* is used in the validation and test splits.

	foldable	rollable	squeezable	containment	tearable
frequency distribution	2	575	156	178	1
affordance	0	1	1	1	0

Table 12: Example of the action frequency distribution and multi-label affordance targets for the object "bottle".

The objects belonging to each affordance and their division into the two sets are presented in Table 13.

Affordance	Set A	Set B
Foldable	'paper', 'mat', 'book', 'bag', 'blanket', 'handkerchief', 'wallet', 'letter', 'towel'	'sock', 'cloth', 'napkin', 'kerchief', 'envelope', 'newspaper', 'shirt'
Rollable	'marble', 'tube', 'tape', 'battery', 'tomato', 'lemon', 'crayon', 'ballpen', 'plastic container', 'lipstick'	'bottle', 'tumbler', 'pencil', 'box', 'cap', 'jar', 'marker', 'can', 'container', 'pen'
Squeezable	'sponge', 'paper', 'plastic bag', 'bag', 'tube', 'ball', 'lemon', 'something', 'wallet', 'toothpaste'	'tissue', 'bottle', 'pillow', 'plastic'
Containment	'bowl', 'mug', 'glass', 'bag', 'basket', 'wallet', 'something', 'pot', 'plate', 'plastic container', 'cup'	'bottle', 'tumbler', 'box', 'vessel', 'cap', 'jar', 'can', 'container'
Tearable	'letter', 'paper'	'envelope', 'tissue', 'newspaper', 'leaf'

Table 13: The objects belonging to each affordance and their division into Sets *A* and *B*.

The following experiments consist of two stages. In the initial stage, the OAcE encoder is trained using object crops as inputs and video representations from Video MAE as targets. To accelerate this process, both the Image Encoder and the Video Encoder modules' representations

are pre-extracted, as only the Mapping Module undergoes training. In the second stage, the trained encoder is tested in affordance categorization using the multi-label affordance targets outlined above.

In the next section, we present a simple method to test whether the video representations extracted from the Video MAE encoder can enhance the affordance categorization performance of an image encoder. In Chapter 5, we use this dataset to evaluate the representations extracted by a Slot Attention model [67], which processes images in an object-centric manner and does not require the object bounding boxes provided by the Something-Else dataset.

4.4 Proposed Method

4.4.1 Object Action-centric Encoder

The architecture and training method of the the Object Action-centric Encoder (OAcE) is depicted in Figure 29. The OAcE takes as input object crops, which are extracted using the bounding box annotations from the Something-Else dataset. The videos of actions belong to the Something's Affordance subset. For every object crop input, the OAcE is trained to produce a representation vector in the action-centric enhanced representation space.

Model	Patch Size	Embed Dim	Depth	Num Heads	Pretrained Source
Video MAE ViT-S	16	384	12	6	[109]
CLIP ViT-B	32	512	12	8	[86]
Image MAE ViT-B	32	512	12	8	[45]

Table 14: Pre-trained model specifications for Video MAE, CLIP, and Image MAE.

Teacher encoding. The teacher model is the pre-trained ViT-S Video MAE, provided by [109]. The model specifications are presented in Table 14. Let us denote the N videos in the Something's Affordances dataset as $\mathcal{X}_t^{(i)}, i \in [1..N]$, each one consisting of $T^{(i)}$ frames (Equation 48). The frames have a fixed height of 224 pixels, but variable width. Before the frames are introduced to the Video MAE, they are pre-processed to a fixed 224×224 shape ($H = W = 224$). In addition, the videos are temporally downsampled to a length of 16 frames, which we refer to as video clips (Equation 49). The Video MAE processes the video clips, and the action-centric representations $\mathcal{F}^{(i)}$, are obtained by average pooling the encoder's token representations. The size of these representation vectors is $d_t = 384$.

$$\textbf{Videos: } \mathcal{X}_t^{(i)} = \{\mathbf{x}_1, \ldots, \mathbf{x}_{T^{(i)}}\} \in \mathbb{R}^{T^{(i)} \times H \times W \times 3} \tag{48}$$

$$\textbf{Video Clips: } V_t^{(i)} = \{\mathbf{v}_1, \ldots, \mathbf{v}_{16}\} \in \mathbb{R}^{16 \times H \times W \times 3} \tag{49}$$

$$\textbf{Teacher representations: } \mathcal{F}(i) = \varphi_{teacher}(V_t^{(i)}) \in \mathbb{R}^{d_t} \tag{50}$$

Image Encoding. Two pre-trained image encoders were used in this experimental section, a CLIP [86] and an Image MAE [45] (Table 14). The Image MAE was fine-tuned in the images of the Something's Affordances dataset for 100 epochs.

As mentioned previously, to reduce interference, the object crops, $C_t^{(i)}$ (Equation 51), are extracted from the first 10 frames of the videos, using the bounding boxes from the Something-Else dataset's annotations. The object crops are then processed with the specified pre-processor

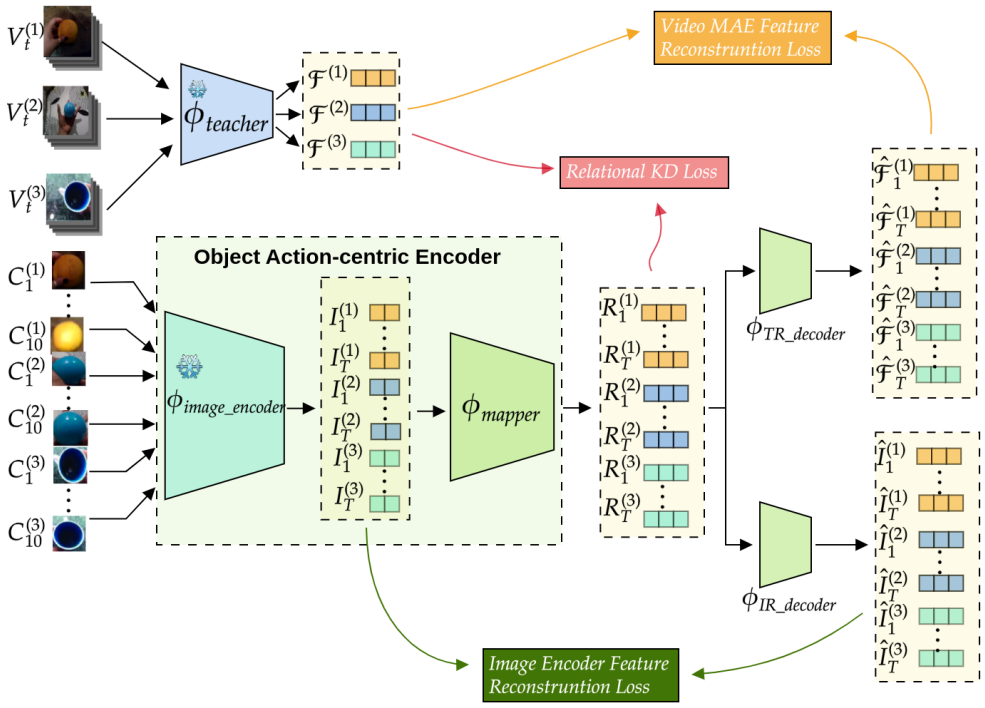


Figure 29: Object Action-Centric Encoder: Knowledge Distillation process and architecture

of each image encoder. Both encoders accept images of shape $H \times W \times 3$, where $H = W = 224$. The image encoder processes the object crops to produce the image representations $I_t^{(i)}$. In both cases, the size of the image representation vectors is $d_i = 512$.

$$\textbf{Object crops: } C_t^{(i)} = \left\{\mathbf{C}_1^{(i)}, \ldots, \mathbf{C}_T^{(i)}\right\} \in \mathbb{R}^{10 \times H \times W \times 3} \tag{51}$$

$$\textbf{Image representations: } I_t^{(i)} = \phi_{image\_encoder}(C_t^{(i)}) \in \mathbb{R}^{d_i} \tag{52}$$

Mapping to the Action-centric Representation Space. The mapping from the image representations, $I_t^{(i)}$, to the OAcE representations, $R_t^{(i)}$, is produced by an MLP (Equation 53) consisting of the following modules, connected sequentially:

1. A linear layer with input size 512 and output size 384
2. A ReLU activation layer
3. A dropout layer
4. A linear layer with input size 384 and output size 384

$$\textbf{OAcE representations: } R_t^{(i)} = \phi_{mapper}(I_t^{(i)}) \in \mathbb{R}^{d_i} \tag{53}$$

Tests involving MLPs with additional layers were carried out, and the outcomes are detailed in the ablation section.

Feature level decoding. To train the OAcE the action-cetric representations are decoded to reconstruct the Video MAE representations (Equation 54)and the Image representations (Equation 55). Targeting both features helped enhance the Image Encoder's capabilities and led to better results than having the OAcE target only the features of the Video MAE.

$$\textbf{Teacher representation (TR) reconstructions: } \hat{\mathcal{F}}_t^{(i)} = \varphi_{TR\_decoder}(R_t^{(i)}) \in \mathbb{R}^{d_t} \tag{54}$$

$$\textbf{Image representation (IR) reconstructions: } \hat{I}_t^{(i)} = \varphi_{IR\_decoder}(R_t^{(i)}) \in \mathbb{R}^{d_i} \tag{55}$$

Losses. The Mapper Module and the two decoder are optimized using three distinct loss functions:

1. **Teacher representations reconstruction loss**: This is the Mean Squared Error (MSE) Loss calculated between the target representations from the Video MAE for each video and the reconstructed representations of the corresponding object's crops in the same video. For a batch B with N video samples:

$$L_{TR} = \frac{1}{N \cdot d_t} \sum_{i=1}^{N} \sum_{t=1}^{10} L_{MSE}(\mathcal{F}^{(i)}, \hat{\mathcal{F}}_t^{(i)}) \tag{56}$$

2. **Image representations reconstruction loss**: This is the Mean Squared Error (MSE) Loss calculated between the image representations from the Image Encoder and the reconstructed image representations. For a batch B with N object crop samples:

$$L_{IR} = \frac{1}{N \cdot d_t} \sum_{i=1}^{N} \sum_{t=1}^{10} L_{MSE}(I_t^{(i)}, \hat{I}_t^{(i)}) \tag{57}$$

3. **Relational KD loss**: This is the Angle-wise Relational Knowledge Distillation Loss (RKD-A) as proposed in [79]. This loss is complementary to the previous feature-based losses and improves the model's performance by focusing on inter-sample relationships. For a triplet of samples, the angle-wise relational potential quantifies the angle created by the three samples in a feature space:

$$\begin{aligned} \psi_A(R_i, R_j, R_k) &= \cos \angle R_i R_j R_k = (\mathbf{e}_{ij}, \mathbf{e}_{kj}) \\ \textit{where} \quad \mathbf{e}_{ij} &= \frac{t_i - t_j}{||t_i - t_j||_2}, \quad \mathbf{e}_{kj} = \frac{t_k - t_j}{||t_k - t_j||_2}. \end{aligned} \tag{58}$$

   The RKD-A loss measures the difference in angle-wise potential between the OAcE representations and the teacher representations:

$$\text{RKD-A} = \frac{1}{|C^3|} \sum_{(C_i, C_j, C_k) \in C^3} L_{MSE}(\psi_A(R_i, R_j, R_k), \psi_A(\mathcal{F}_i, \mathcal{F}_j, \mathcal{F}_k)) \tag{59}$$

   To restrict the increase in computational complexity that this loss introduces, C^3 is a set of 50 triplets, randomly selected from each batch.

The total loss is the sum of the above losses:

$$L = L_{TR} + L_{IR} + \text{RKD-A} \tag{60}$$

Ablation tests and experiments with the individual losses being combined in different ways are presented in the loss function ablation section.

Training. All models were trained for 20 epochs with the use learning rate scheduler and the Adam optimizer [56]. The learning rate scheduling involved a two-phase approach [3]: an initial linear warm-up until $lr = 0.001$ that lasts for 5% of the total training steps, followed by exponential decay. This approach aimed to stabilize the training process and improve convergence.

4.4.2 Evaluation of representations

The evaluation was carried out using two methods, both tested on the multi-label affordance targets of the Something's Affordances dataset: (i) linear probing and (ii) training an MLP classification head on top of the frozen OAcE representations. The aim of the experimental evaluation is to test whether the OAcE encoder can enhance two state-of-the-art image representations: CLIP and Image MAE. It is important to note that the CLIP representations have not been trained on the dataset used for evaluation, whereas the Image MAE has been fine-tuned on this dataset.

All the performance metrics were **macro-averaged**, treating all the affordances equally regardless of the number of samples present in them.

The evaluated representations are as follows:

1. **GT:** Ground truth was created by training classifiers on the target teacher data, as if the model had access to "perfect" memories of the actions associated with each object. This highlights the valuable signal in the teacher model's representations.

2. **OAcE on CLIP:** Training classifiers on the OAcE representations, with CLIP as the image encoder.

3. **CLIP:** Training classifiers on the CLIP representations.

4. **OAcE on IMAE:** Training classifiers on the OAcE representations, with Image MAE as the image encoder.

5. **IMAE:** Training classifiers on Image MAE representations.

6. **OAcE + IMAE:** Training classifiers on concatenated representations of Image MAE and OAcE (on IMAE).

Linear Probing. Linear probing has been used as a representation evaluation protocol in a variety of studies, including [86, 45]. It involves training a linear classifier on top of the representations. In the context of the Somethig's Affordance dataset, multi-label classification requires the training of five binary linear classifiers - one for each affordance. The classifier chosen for this experimental section was Logistic Regression, which is a generalized linear model. The Scikit-learn [82] implementation of the L-BFGS-B [11] large-scale bound-constrained optimization algorithm was used to train the classifiers on the data. The results presented in Tables 15 and 16.

Configuration	Recall	Precision	F1 Score	Accuracy
GT	0.7508	0.9444	0.8349	0.8774
OAcE on CLIP	**0.7275**	**0.9224**	**0.8116**	**0.8610**
CLIP	0.7217	0.9147	0.8050	0.8561
OAcE on IMAE	0.6514	0.8986	0.7512	0.8256
IMAE	0.6863	0.8942	0.7740	0.8364
OAcE + IMAE	0.6964	0.8973	0.7821	0.8411

Table 15: Linear Probing performance metrics for Video-based split of the Something's Affordance dataset

Configuration	Recall	Precision	F1 Score	Accuracy
GT	0.6256	0.6240	0.5543	0.6845
OAcE on CLIP	**0.6360**	**0.6404**	**0.5707**	**0.6980**
CLIP	0.6256	0.6240	0.5543	0.6845
OAcE on IMAE	0.5575	0.5853	0.4821	0.6341
OAcE + IMAE	0.5994	0.5931	0.5341	0.6722
IMAE	0.5984	0.5907	0.5301	0.6681

Table 16: Linear Probing performance metrics for Object-based split of the Something's Affordance datase

MLP Classification Head. Linear probing, is a useful evaluation protocol, but it misses the opportunity to learn from strong but nonlinear representation spaces. To further test the OAcE representations in that respect, a small-scale MLP head was evaluated on the same task. The classifications head architecture is the following:

1. Linear layer (input d_t = 384,output = 1024)
2. Relu activation layer
3. Linear laye (input: d_t = 1024, output = 5)
4. Sigmoid activation on each output

The neural network training followed a similar approach to the OAcE Mapper module, using the Adam optimizer [56] and a learning rate scheduler with a maximum learning rate of 0.001. To classify the results, thresholding is applied to the outputs of the classifier's last layer, which fall within the [0, 1] range due to the sigmoid activation. The thresholding is tuned on the validation set, on each one of the five heads separately, to maximize the F1 score of the classifier. The experimental results are presented in Tables 17 and 18. Additionally, the Figures 30 and 31 show the F1 score different affordance labels.

Configuration	Recall	Precision	F1 Score	Accuracy
GT	0.8265	0.9380	0.8776	0.9045
OAcE on CLIP	**0.8467**	**0.8782**	**0.8611**	**0.8878**
CLIP	0.8195	0.8858	0.8505	0.8817
OAcE on IMAE	0.8138	0.8173	0.8145	0.8493
AcE + IMAE	0.8051	0.8331	0.8174	0.8538
IMAE	0.7785	0.8359	0.8046	0.8458

Table 17: MLP Classification performance metrics for the Video-based split of the Something's Affordances dataset

Configuration	Recall	Precision	F1 Score	Accuracy
GT	0.6900	0.7737	0.7080	0.8058
OAcE on CLIP	**0.6209**	**0.7191**	**0.6562**	**0.7688**
CLIP	0.6154	0.6912	0.6304	0.7472
AcE on IMAE	0.5271	0.6574	0.5656	0.7136
AcE + IMAE	0.5501	0.6656	0.5838	0.7218
IMAE	0.5410	0.6633	0.5763	0.7186

Table 18: MLP Classification performance metrics for the Object-based split of the Something's Affordances dataset

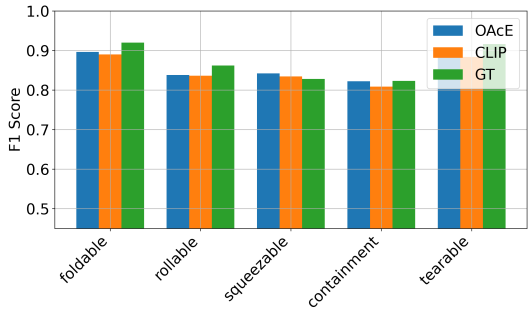


Figure 30: F1 score for the different affordance labels when tested on the Video-based split.

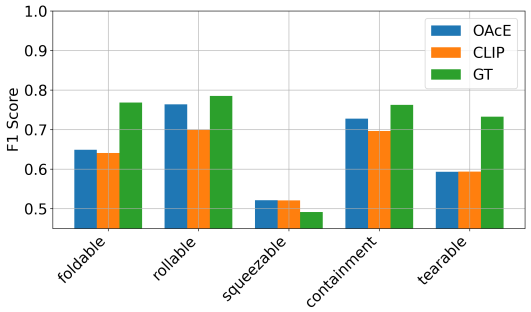


Figure 31: F1 score for the different affordance labels when tested on the Object-based split.

MLP Depth Ablations. We tested for the impact of varying MLP depths on the performance metrics. The top performance metrics were achieved using a two-layer MLP, as presented earlier and the performance degrades with the increase of depth as seen in Figures 32 and 33 where the width of the layers is presented in Table 19.

# Layers	Layers	# Learnable Parameters
2	[512, 384]	804,608
3	[1024, 512, 384]	1,592,064
4	[2048, 1024, 512, 384]	4,215,552
5	[1024, 2048, 1024, 512, 384]	5,789,440
6	[512, 1024, 2048, 1024, 512, 384]	6,052,096

Table 19: MLP Architectures and Learnable Parameters

	Accuracy	Precision	Recall	F1 Score
All Losses	0.8868	0.8834	0.8370	0.8586
L_{TR} + RKD-A	0.8878	0.8782	0.8467	0.8611
L_{TR}	0.8850	0.8801	0.8355	0.8567
L_{TR} + L_{IR}	0.8875	0.8845	0.8371	0.8597
RKD-A	0.8759	0.8644	0.8289	0.8454

Table 20: Loss functions ablation results for the OAcE on CLIP configuration tested on the Video-based split.

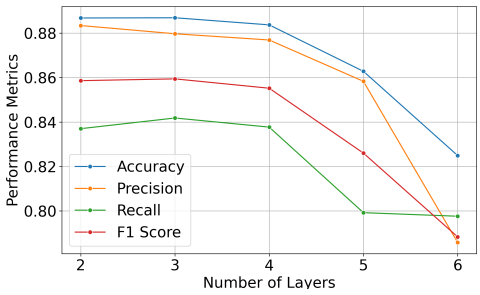


Figure 32: Performance Metrics for different MLP depths tested on the Video-based split.

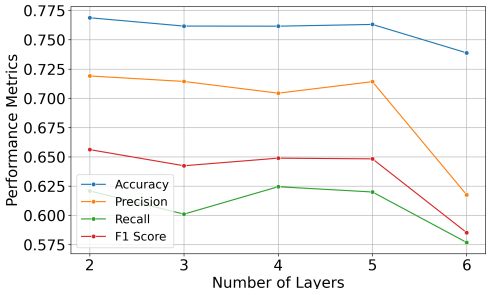


Figure 33: Performance Metrics for different MLP depths tested on the Object-based split.

Loss function Ablations. Tables 20 and 21 show the results of the ablation study on the three previously discussed loss functions. In the Video-Based split the difference is negligible between the different losses. In the Object-based split the relative loss, RKD-A, produces good results alone, while the incrorporation of the other two losses marginally increases the performance. We further tested all the losses computing the L_1 and *Smooth_L1* loss (in place of the *MSE*). These degraded the performance of the representations.

Qualitative Results. Figure 34 presents a collection of successful examples from the test set of the object-based split, using the OAcE on CLIP model. One important observation is the models ability to classify objects from blurry, obstructed or incomplete crops. We hypothesize that this capability can be credited to the fact that the training set is derived from real-world videos. Figure 35.

	Accuracy	Precision	Recall	F1 Score
All Losses	0.7688	0.7191	0.6209	0.6562
L_{TR} + RKD-A	0.7658	0.7097	0.6189	0.6508
L_{TR}	0.7462	0.6825	0.6050	0.6240
L_{TR} + L_{IR}	0.7553	0.6985	0.6167	0.6411
RKD-A	0.7652	0.7190	0.6082	0.6461

Table 21: Loss functions ablation results for the OAcE on CLIP configuration tested on the Object-based split.

Figure 34: Affordance Categorization examples using the OAcE on CLIP model, on the test set of the object-based split.

4.5 Observations

The results in **Tables 15-18** indicate that in general the ground truth Video MAE representations demonstrate better performance compared to the image encoders . This improvement could be attributed to the fact that the image encoders are trained on out-of-domain data while the Video MAE has had some training on the SSv2 dataset. To address this we fine-tune the Image MAE to images from the dataset. However, the CLIP encoder still outperforms the Image MAE and a more conclusive result would require the fine-tuning of a CLIP model using the SSv2 dataset.

However, we consider the fact that Video MAE representations demonstrate better performance an indication that there is useful signal in the Video MAE representations and these experiments showcase our attempt to utilize it. These ground truth representations appear to have significant non-linear components, as their performance improves significantly in the MLP classification.

The results presented in Figures 30 and 31 show that when evaluating using the Video-based split all the affordance labels present the same difficulty for the models. On the other hand, when using the Object-based split the affordance *squeezable* demonstrates the worst scores. Notably,

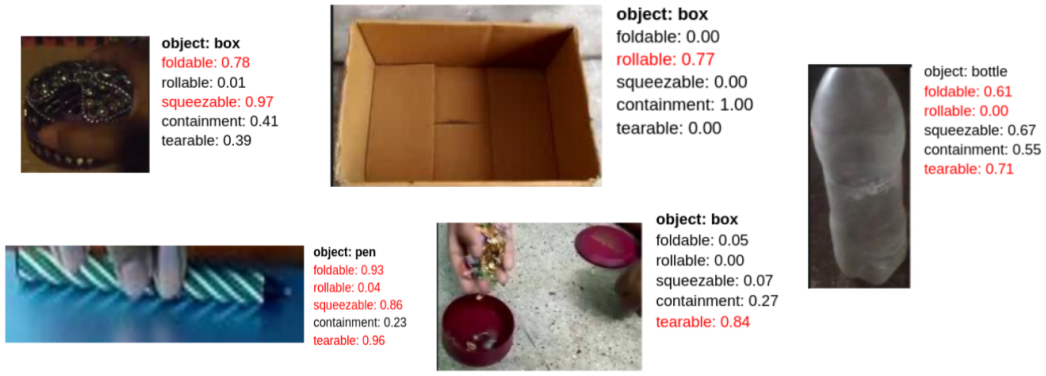


Figure 35: Unsuccessful Affordance Categorization examples using the OAcE on CLIP model, on the test set of the object-based split.

it is the only instance where the Video MAE GT representations perform worse compared to the others. This is likely due to the different characteristics of the objects in *Set A* and *Set B* (Table 13) that lead to more challenging generalization.

Overall the OAcE proposed method provides a marginal improvement to the image encoders. This improvement is more noticeable in the object-based split of the dataset. This object-based split presents a bigger challenge to the models, as it introduces unseen objects in the test set, requiring the models to perform compositional generalization. In total, the OAcE on CLIP shows better performance, closely approaching the model trained on ground-truth representations. In the case of Image MAE, OAcE did not always provide improvements on its own in some experiments. However, concatenating the OAcE representations with the image representations (OAcE + IMAE) resulted in improvements.

4.6 Limitations and future directions

Evaluation Method Limitations. The current evaluation is limited by a small dataset with few categories. In future work, more affordance categories could be extracted from this dataset. A more comprehensive evaluation would involve using larger datasets like Ego4D [36] and EPIC-Kitchens [17], which include object annotations. However, a significant challenge would be the training of the Video MAE ViT, or an alternative teacher model, from scratch on these larger datasets, given their extended video durations and higher quality.

Additionally, as noted before, in our experiments the CLIP image encoder outperforms the Image MAE, even though the CLIP encoder is only trained on out-of-domain data. To strengthen the case for the action-to-object representation method, particularly for CLIP, it is necessary to compare it with a fine-tuned model. However, since no official code has been released for CLIP training, its fine-tuning on the SSV2 dataset based on the details provided in the paper is postponed for later research.

Architecture limitations. One limitation of the OAcE is its dependence on an object detection or segmentation module (e.g. YOLO [89], SAM [57], EgoHOS [123], Mask R-CNN [44]) to extract the object crops from the videos and supply the encoder.

During the initial experimentation phase, an automated visual extractor was developed and tested. This visual extractor used pre-trained hand-object segmentation models from [123]. The

authors propose a Hand-Object Segmentation procedure that uses three segmentation models sequentially. The first one segments the hands, the second one takes the hand mask and the image and produces the mask of the hand-object contact boundary, and the third one takes the contact boundary and the image and produces the masks of the interacting objects. The masks from the last stage were used to produce the bounding boxes for the interacting objects. However, in the Something-Something v.2 dataset, the extracted crops were mostly unsuccessful. We assume this is because the model was trained on egocentric data while Something-Something v.2 videos have varying perspectives.

As a result, it was decided to use the hand-crafted annotations of the Something-Else dataset for this chapter's proof-of-concept and then focus on a representation learning framework that automatically extracts the objects, and object-centric representations from a scene. This framework is Slot Attention, and the next chapter documents an attempt to understand the basic principles and evaluate the model's representations in a novel setting.

4.7 Conclusion

In this section, we experimented with an action-to-object distillation process that attempts to transfer the knowledge of a Video MAE that models the videos of actions of the Something Something dataset to object-centric image encoders. The representations of the method were evaluated using two approaches, both tested on the multi-label affordance targets of the Something's Affordances dataset: (i) linear probing and (ii) training an MLP classification head on top of the frozen representations. The experiments show that the methods produce a marginal yet consistent enhancement to the representations.

An interesting future direction could include larger scale model implementations and datasets. Additionally, a promising future direction could include associating the agents' own experiences with the objects, possibly with an online KD framework. Overall, these methods show promise in developing vision systems that provide agents with a head start in understanding the agent-object interactions of the real world.

5 Slot Attention Representations

5.1 Introduction

One limitation of the previous encoder was its dependence on an object detection or segmentation module (e.g. YOLO [89], SAM [57], EgoHOS [123]) to supply the encoder with object image crops from a scene. This experiment focuses on a technique that can address that limitation by achieving automatic segmentation of an image or video to objects[6]. This technique is centered on the Slot Attention [67] architecture and falls into the category of object-centric representation learning.

In this experiment, the object representations are drawn from the *SOLV* model, which was presented in the paper "Self-supervised Object-Centric Learning for Videos" [3].This model successfully achieves multi-object segmentation in real-world videos (Figure 36), and in the process extracts representations for each of the objects in the input. Using the *Something's Affordance* dataset, we aim to evaluate these representations in the task of affordance categorization. Before discussing the experimental methodology and results, some of the most important elements of the *SOLV* model are presented in the following sections.

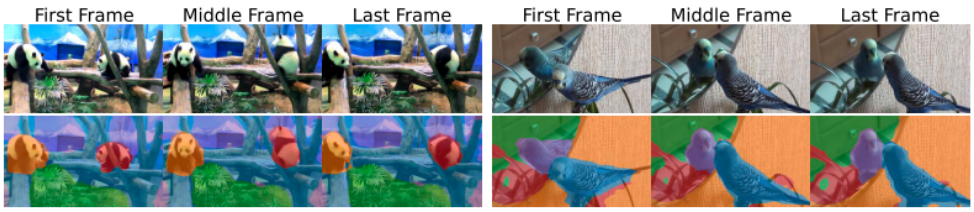


Figure 36: SOLV: Instance segmentation results of first, middle, and last frames of videos on the Youtube-VIS-2019 dataset. Source: [3]

5.2 Theoretical Background

5.2.1 Slot Attention

This section presents key aspects of the Slot Attention module, introduced by Locatello et al. in their paper "Object-Centric Learning with Slot Attention" [67]. Slot Attention is a differentiable interface, based on iterative dot product attention, that can be used to bind objects from a visual input to a set of variables known as slots. As a result, it allows for a more structured representation of the input scene. To better understand the Slot Attention module, let us consider an image input that has been encoded into N feature vectors with positional encoding, each of size D_{inputs}. The module processes these feature vectors to produce K slots, each of size D_{slots}. For each iteration, the input and slot vectors are first transformed to keys and queries, using the learnable linear projections k, q. The attention matrix is computed as follows:

$$\text{attn}_{i,j} := \frac{e^{M_{i,j}}}{\sum_{l=1}^{K} e^{M_{i,l}}} \quad \text{where} \quad M_{i,j} := \frac{1}{\sqrt{D}} k(\text{inputs}) \cdot q(\text{slots})^T \in \mathbb{R}^{N \times K}, \quad i \in \{1, \dots, N\}, \quad j \in \{1, \dots, K\} \tag{61}$$

[6]The term "object" in this setting takes a more general meaning, referring to a semantic component of a visual input (image or video).

It is important to note that the softmax operation is applied across the slot axis. This allows for information exchange between the slots which compete for attending to each feature vector. Subsequently, the *value* vectors are produced through the learnable v linear projection, and the slot update vectors are computed by taking the weighted mean [7] of the inputs, using the attention vectors of each slot as weights:

$$\text{updates} := W^T \cdot v(\text{inputs}) \in \mathbb{R}^{K \times D} \quad \text{where} \quad W_{i,j} := \frac{\text{attn}_{i,j}}{\sum_{l=1}^{N} \text{attn}_{l,j}}. \tag{62}$$

The updates are then used to update the slots values through a Gated Recurrent Unit (GRU) [15]. Using these new slot values, the process is repeated T times, as described in the pseudo-code of Algorithm 2.

Algorithm 2 Slot Attention Module. Source: [67]

Input: **inputs** $\in \mathbb{R}^{N \times D_{\text{input}}}$, **slots** $\sim \mathcal{N}(\mu, \text{diag}(\sigma)) \in \mathbb{R}^{K \times D_{\text{slots}}}$
Layer params: **k**, **q**, **v**: linear projections for attention; GRU; MLP; LayerNorm (x3)
inputs = LayerNorm(**inputs**)
for $t = 0 \ldots T$ **do**
    slots_prev = **slots**
    slots = LayerNorm(**slots**)
    attn = Softmax$\left(\frac{\mathbf{k}(\mathbf{inputs}) \cdot \mathbf{q}(\mathbf{slots})^\top}{\sqrt{D}}\right)$, axis='slots' `# norm. over slots`
    updates = WeightedMean(weights=**attn** + ϵ, values=**v**(**inputs**)) `# aggregate`
    slots = GRU(states=**slots_prev**, inputs=**updates**) `# GRU update (per slot)`
    slots += MLP(LayerNorm(**slots**)) `# optional residual MLP (per slot)`
end for
return slots

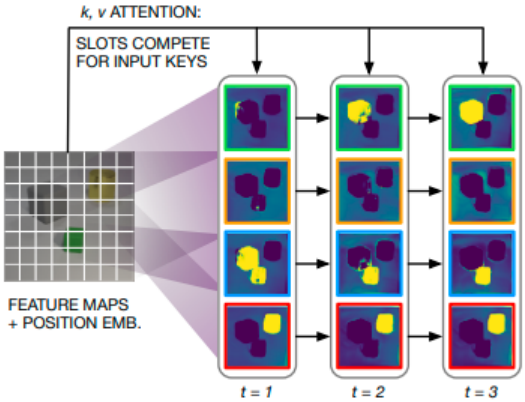


Figure 37: Slot Attention module. Source:[67]

Locatello et al. [67] use a shared set of Gaussian mean and variance parameters to initialize all slots. This can improve the generalization capabilities of the model, as any slot can bind

[7] Locatello et al. [67] have empirically shown that taking the weighted mean, rather than the weighted sum, improves training stability.

to any object in the input. However, as discussed in their paper, this approach can also degrade the performance of the model in some tasks. In contrast, the model used in the following experiments, SOLV, uses per-slot parameterization. This method allows each slot to specialize in a specific type of object, and such initialization enables the application of attention to the instances of each object through time (temporal binding). Per-slot parameterization and initialization can also be reminiscent of the mixture-of-experts (MoE) [12] paradigm, by encouraging each slot to specialize on specific characteristics and promote inter-slot collaboration to extract useful representations about a scene.

The Slot Attention module is followed by a *Spatial Broadcast Decoder* as described by Watters et al.[113]. This decoder processes each slot individually producing K outputs of shape $H \times W \times 4$. The first three dimensions correspond to the RGB color channels, while the last dimension represents an alpha mask. In computer graphics, the alpha layer is used to represent the transparency of an image [84]. In this case, the alpha masks are normalized using the softmax operation over the slots, and then used as weights to combine the slots to the reconstructed image. This reconstructed image can then be used to compute the reconstruction loss used to train the model. This is another point in the model's pipeline where slots exchange information through the simulated competition induced by the softmax function - slots achieve a higher alpha score at the parts of the image that they can reconstruct better.

In general, Slot Attention is characterized by the following two very important properties:

- Permutation invariance with respect to the input: The output is independent of permutations applied to the input.
- Permutation equivariance with respect to the order of the slots: When permuting the slots, output permutes correspondingly.

More formally, let π_i, π_s be the permutation matrices that represent the permutation of the inputs and the slots, respectively. Then:

$$\text{SlotAttention}(\pi_i \cdot \text{inputs}, \pi_s \cdot \text{slots}) = \pi_s \cdot \text{SlotAttention}(\text{inputs}, \text{slots})$$

In the setting of object and multi-object recognition, these invariances are crucial as they produce more coherent representations. Specifically, the permutation invariance property ensures that similar slot representations are computed from a scene when the objects are rearranged. Any difference in slot representations will only be due to the different positional embeddings. The next section discusses an enhancement to the Slot Attention architecture that establishes object pose invariance.

5.2.2 Invariant Slot Attention

This section presents key aspects of the Invariant Slot Attention (ISM) architecture (Figure 38), which was introduced by Biza et al. in the paper "Invariant Slot Attention: Object Discovery with Slot-Centric Reference Frames" [8]. This architecture aims at processing the visual signal in a way that disentangles object appearance from object pose (position, orientation, and scale). Inspired by the evidence that the brain processes objects by computing both egocentric and allocentric [8] reference frames and representations [10], ISM applies positional encoding to the input vectors based on each slot's relative reference frame.

The authors of the ISA paper [8] tested the performance of the framework by isolating and combining the three pose invariances: translation, scale, and rotation. They found that the

[8] Allocentric: independent of the subject's point of view.

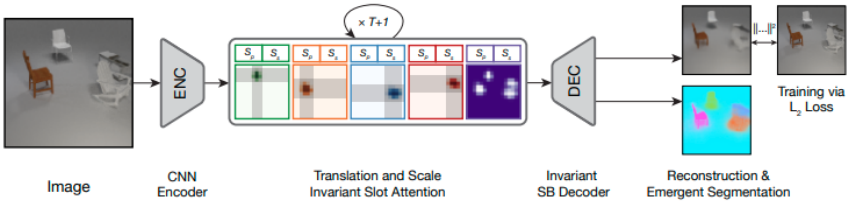


Figure 38: Ivariant Slot Attention. Source: [8]

framework that consistently delivered the best results was the Translation and Scaling Invariant Slot Attention (ISA-TS). The ISA-TS algorithm computes the absolute and relative reference frames using position encoding. In this implementation, the position encodings are 2D grids scaled to $[-1, 1]$, with each grid cell corresponding to an image token. The relative grid is computed from by shifting and scaling the absolute grid using each slot's attention weights (Figure 39 and Equation 63).

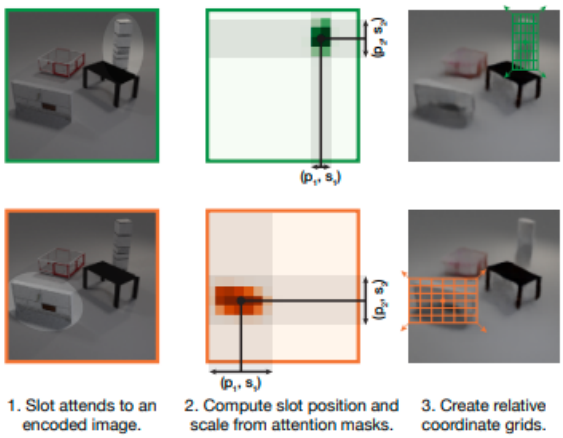


Figure 39: Computation of the relative coordinate grids by the Translation and Scaling Invariant Slot Attention (ISA-TS) module. Source: [8]

$$\text{rel\_grid}^{(k)}_{ISA-TS} = \frac{\text{abs\_grid} - S_p^{(k)}}{S_s^{(k)}} \tag{63}$$

The slot positions, $S_p^{(k)}$, are calculated as the center mass of the attention weights for each slot (Equation 64). The slot scales, $S_s^{(k)}$, are the weighted standard deviation from the slot positions, using the attention weights (Equation 65).

$$S_p^{(k)} = \frac{\sum_{n=1}^{N} \text{attn}_n^{(k)} * \text{abs\_grid}_n}{\sum_{n=1}^{N} \text{attn}_n^{(k)}} \tag{64}$$

$$S_s^{(k)} = \sqrt{\frac{\sum_{n=1}^{N} (\text{attn}_n^{(k)} + \epsilon) * (\text{abs\_grid}_n - S_p^{(k)})^2}{\sum_{n=1}^{N} (\text{attn}_n^{(k)} + \epsilon)}} \tag{65}$$

Finally, *keys* and *values* are computed from the input vectors as follows:

$$\text{keys}^{(k)} = f\left(K(\text{inputs}) + g(\text{relative\_grid}^{(k)})\right) \tag{66}$$

$$\text{values}^{(k)} = f\left(V(\text{inputs}) + g(\text{relative\_grid}^{(k)})\right) \tag{67}$$

That means that during the slot computation, in each iteration, the algorithm computes $N \times K$ *keys* and *values*, and K *queries*. This method results in a small increase in computational complexity as in plain Slot Attention, *keys* and *values* are N and are computed using the image's absolute grid.

In order to incorporate rotation invariance into the above framework (ISA-TSR), the object's orientation is estimated by applying Principal Component Analysis [119] to the absolute grid weighted by the attention mask of each slot. This computes the axes with the highest variation for the object that each slot is attending to (Equation 68).

$$v_1^{(k)}, v_2^{(k)} = \text{PCA}(\text{w\_abs\_grid}^{(k)}), \quad \textit{where} \quad \text{w\_abs\_grid}^{(k)} = \text{attn}^{(k)} \odot \text{abs\_grid} \tag{68}$$

The axes are further processed ($\tilde{v_1}^{(k)}, \tilde{v_2}^{(k)}$) in order to avoid mirroring the grids and rotating more than 45°. The relative grids are then computed as described in Equation 69.

$$\text{rel\_grid}^{(k)}_{ISA-TSR} = \left(S_r^{(k)}\right)^{-1}\left(\text{abs\_grid} - S_p^{(k)}\right)/S_s^{(k)}, \quad \textit{where} \quad S_p^{(k)} = \begin{bmatrix} | & | \\ \tilde{v_1}^{(k)} & \tilde{v_2}^{(k)} \\ | & | \end{bmatrix} \tag{69}$$

In [8], the ISA-TSR framework produced mixed results compared to ISA-TS. The latter is integrated into the SOLV model, which is presented in the next section.

5.2.3 Self-supervised Object-Centric Learning for Videos (SOLV)

This section presents key aspects of the SOLV model [3] that is the main focus of this experimental part. This model was introduced by Aydemir et al. in 'Self-supervised object-centric learning for videos'. The goal of this model is to discover, track and segment objects from video inputs of complex real-world scenes. The SOLV framework (Figure 40) achieves this by implementing spatial-temporal slot attention. At first, each frame is passed through the ISM bottleneck, where the semantic components and their representations are computed. Subsequently, each slot is enhanced with temporal information by attending to the corresponding slots in neighboring frames. The model is trained as a masked autoencoder, reconstructing the central frame of the input video clip at the dense feature space level, provided by the DINOv2 feature extractor [75]. The following paragraphs present further details about the SOLV architecture's modules.

The *Visual Encoder* is the first module in the SOLV's pipeline. It takes as input a video clip of $2n + 1$ frames (Equation 70), divide each frame to $N = HW/P^2$ non-overlapping patches of size P (Equation 71), applies token drop (during the training phase) (Equation 72), and encodes each token using the frozen DINOv2 pre-trained ViT [75] (Equation 73). It is important to note that while the SOLV model utilizes ISA, the DINOv2 tokens contain positional information, as the ViT encoder adds positional embeddings to the patches.

$$\textbf{Video Clip: } \mathcal{X}_t = \{\mathbf{x}_{t-n}, \ldots, \mathbf{x}_t, \ldots, \mathbf{x}_{t+n}\} \in \mathbb{R}^{(2n+1)\times H\times W\times 3} \tag{70}$$

$$\textbf{Tokenized Video Clip: } \mathcal{V}_t = \{\mathbf{v}_{t-n}, \ldots, \mathbf{v}_t, \ldots, \mathbf{v}_{t+n}\} \in \mathbb{R}^{(2n+1)\times N\times (3P^2)} \tag{71}$$

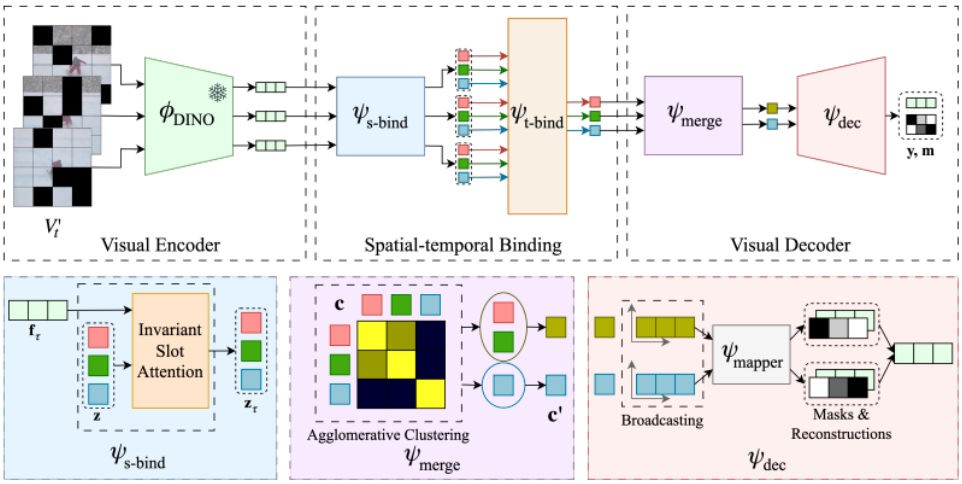


Figure 40: The SOLV architecture. Source: [3]

Tokenized Video Clip after Dropout: $\mathcal{V}'_t = \{\text{drop}(\mathbf{v}_{t-n}), \ldots, \text{drop}(\mathbf{v}_{t+n})\} \quad \in \mathbb{R}^{(2n+1)\times N'\times(3P^2)}, \quad N' < N$ (72)

Feature extraction: $\mathcal{F}_t = \{\phi_{DINO}(\mathbf{v'}_{t-n}), \ldots, \phi_{DINO}(\mathbf{v'}_t), \ldots, \phi_{DINO}(\mathbf{v'}_{t+n})\} \quad \in \mathbb{R}^{(2n+1)\times N'\times D}$ (73)

The subsequent module is the *Spatial Binder*, which implements ISA-TS to each frame independently, as presented in the previous section. The encoded tokens are first reduced in dimension, and then are passed to an ISA-TS module that computes the following for each frame:

- Slot vectors: $z^{(k)} \in D_{slot}$
- Slot positions: $S_p^{(k)} \in \mathbb{R}^2$
- Slot scales: $S_s^{(k)} \in \mathbb{R}^2$

In contrast to the original slot attention framework, which uses random slot initialization, in this implementation the initial slot vectors are deterministic static learnable parameters. These parameters are shared across frames, laying the groundwork for the Temporal Binder module. Given that neighboring frames have similar visual information and that the depicted objects do not change drastically from one frame to the next, it is reasonable to assume that in most cases slots with the same index will bind to the same objects across frames.

The *Temporal Binder* is a transformer encoder, which enhances the slot representations by computing their affinities over time. For each slot, the self-attention module processes the $2n+1$ slot vectors, producing a single slot vector that contains temporal information. Additionally, learnable temporal positional encoding is applied to these vectors to utilize the temporal causality signal available in the video data. As a result, the output of this module is a set of slot vectors $c_{temp} \in \mathbb{R}^{K\times D_{slot}}$.

The next module is a *Slot Merger* that dynamically computes the optimal number of slots for each scene and groups the slots (Figure 41) using the Agglomerative Clustering (AC) algorithm [130, 96]. The AC is a bottom-up hierarchical clustering algorithm that begins with each observation (each slot) in a different group and at every iteration merges the two most similar groups according to a chosen distance metric. The distance measure between two vectors used in this implementation is the cosine distance, as described in Equation 74.

$$d_{cos}(a, b) := 1 - \cos(\partial_{a,b}) = 1 - \frac{ab^T}{\|a\| \, \|b\|} \tag{74}$$

Additionally, to compute the distance between two groups the Complete Linkage (CL) clustering method is applied. This method, also known as the furthest-neighbor technique, defines the dissimilarity between two groups A and B as the maximum pairwise distance between two vectors $a \in A, b \in B$ (Equation 75).

$$d_{CL}(A, B) = \max_{\substack{a \in A \\ b \in B}} d_{cos}(a, b) \tag{75}$$

Finally, the AC implementation used in SOLV applies a distance threshold above which slot groups are not merged. It is important to note here that there is an inherent difficulty in evaluating models in the task of object segmentation, as the optimal amount of segmentation granularity might vary for different settings. Methods like this one have the potential to lead to dynamic segmentation and object grouping that might be useful in developing agents that can plan and reason.

Figure 41: SOLV segmentation results before (right) and after (left) the Slot Merger module. Source: [3]

The last module of the SOLV model is a *Spatial Broadcast Decoder* [113] that takes in the reduced number of slot vectors from the merger and reconstructs the central frame's features, y, in a similar was as the one described in the section for the SA framework. The loss function that is minimized during training is the MSE loss of this reconstruction with the DINOv2 features of the central frame of each video clip (Equation 76).

$$\mathcal{L}_{SOLV} = \|\phi_{\text{DINO}}(\mathbf{v}_t) - \mathbf{y}\|^2 \tag{76}$$

In total, the SOLV framework extracts object-centric representations by reconstructing the dense features of a frame while utilizing information from an entire video clip. Interestingly,

object segmentation masks emerge as a by-product of this self-supervised process. In the next section, we test for another by-product of this process — the slot representations — and their utility for the downstream task of affordance categorization.

5.3 Proposed Method

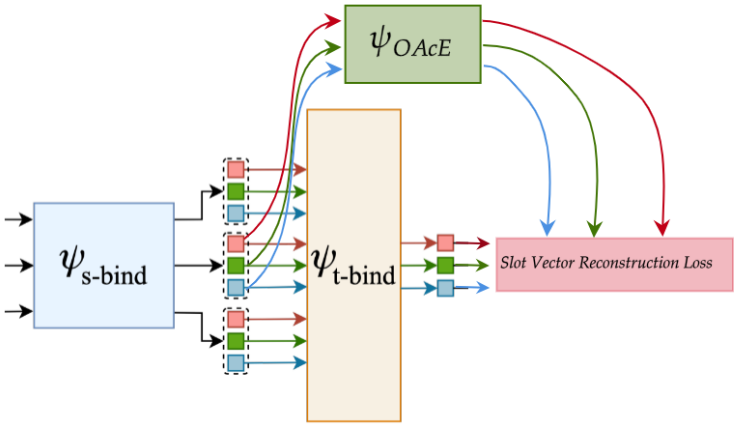


Figure 42: The OAcE training on the spatio-temporal binding module of the SOLV model. Adapted from:[3]

In this experimental section, we evaluate the slot representations of SOLV in the affordance categorization task, using the Something's Affordances (SA) dataset. Unlike the experiments with the Video MAE representations where the OAcE encoder received an object crop as input, the SOLV model is able to process an image of an entire scene and automatically segment it.

As the first step of the representation learning part of the experiment, the SOLV model was fine-tuned on videos of the video-based split of Something's Affordances (SA-Vb) for 100 epochs. The training code was provided by the supplementary material of [3] and the characteristics of the pre-trained SOLV model are presented in Table 22. Due to the complexity of the training in this section, we focus solely on the video-based split of this data set. The object-based split and compositional generalization evaluation are reserved for future work.

The modular design of the SOLV model allows the extraction of object-centric representations and representation learning at various locations in the model pipeline. The two points of focus are the slot vector outputs of the Spatial Binder and the Temporal Binder. It is important to note here that even though the Spatial Binder focuses on the spatial characteristics in a per-frame manner, it has been trained in an overall system that processes temporal information. These spatial slot vectors are optimized to attend to inter-frame slots and thus can be though to include, and be educated by temporal signal. In the following experiments where we compare the video clip informed slot representations from the Temporal Binder with the image slot representations from the Spatial Binder.

Additionally, similarly to the approach taken in the previous chapter, we attempt to associate some information about the actions with the object representation vectors. This is done using an MLP, which receives the Spatial Binder's slot vectors from the central frame of a video clip and is trained to predict the slots generated by the Temporal Binder's output, as shown in Figure 42. This MLP is trained on a dataset of action videos and is referred to as $OAcE_{SOLV}$

Component	Specifications
Feature Extractor (ϕ_{DINO})	ViT-B/14 architecture with DINOv2 pretraining Output: Last block without CLS token Positional embeddings added to patches
Spatial Binding	Projection to slot dimension $D_{\text{slot}} = 128$ 2-layer MLP, layer normalization Invariant Slot Attention (ISA) with GRU cell update Residual MLP with hidden size $4 \times D_{\text{slot}}$ Projection layers (q, k, v) size: D_{slot} Binding operation repeated 3 times
Temporal Binding	Transformer encoder with 3 layers, 8 heads Hidden dimension: $4 \times D_{\text{slot}}$ Temporal positional embedding with normal distribution
Slot Merging	Agglomerative Clustering with complete linkage
Decoder Mapper	5 linear layers with ReLU activations Hidden size: 1024 Final layer maps to dimension of ViT-B tokens + a (768 + 1)

Table 22: Pre-trained SOLV[3] model Specifications.

(Object Action-centric Encoder) in the following section. The $OAcE_{SOLV}$ model has the following architecture:

- **Linear Layer 1**: `Linear(`D_{slot}`,` $4 \times D_{slot}$`)`
- **ReLU Activation**: `ReLU(inplace=True)`
- **Linear Layer 2**: `Linear(`$4 \times D_{slot}$`,` D_{slot}`)`
- **Dropout**: `nn.Dropout(p=0.1)`
- **Residual Connection**: `output += input`

The train split of the SA-Vb dataset contains 62,330 videos. Due to the increased complexity of this training method, a smaller random subset of 306 videos is taken on each epoch, without replacement, using PyTorch's [81] Subset Random Sampler [105]. Training $OAcE_{SOLV}$ involves training for 10 epochs, using batches of size 18 and a learning rate scheduling approach that involves an initial linear warm-up until $lr = 0.0004$, followed by exponential decay. The Smooth L1 Loss function is used to improve robustness.

5.3.1 Evaluation

To evaluate the SOLV representations, we train them in the affordance categorization task on the SA-Vb dataset, using the bounding box annotations of the Somethin-Else dataset. The Affordance Categorization Module (ACM) is an MLP with the following architecture:

- **Batch Normalization Layer** [50]: `BatchNorm1d(`D_{slot}`)`
- **Linear Layer 1**: `Linear(`D_{slot}`, 1024)`
- **ReLU Activation**: `ReLU(inplace=True)`
- **Linear Layer 2**: `Linear(1024, 5)`
- **Dropout**: `Dropout(p=0.1)`
- **Sigmoid Activation**: `Sigmoid()`

The symbols and variables of the training algorithm for one epoch (Algorithm 3) are explained in the list below:

- B: The batch size.
- H and W: The height and width of the images, respectively.
- K: The number of slots used in the SOLV module.
- D_{slot}: The dimension of the slot vectors.
- N: $N = 864$ The number of tokens extracted by the Dino Visual Encoder.
- *slots_of_obj*: The slots that correspond to interacting objects in the images, identified by matching segmentation masks with bounding boxes.
- *slot_nums*: The number of slots that remain in each input after the *Slot Merger Module*. The $slots[i] = \mathbf{0}$ for $i > slot\_nums$
- *find_slots_of_obj*(): Function that finds which slot attends the most to the interacting object of the scene.

The training algorithm for one epoch is the following:

Algorithm 3 Affordance Categorization Module (ACM) - Train One Epoch

```
1: Input: Dataloader, SOLV, OACE_SOLV, ACM, representation_type
2: # images : B × H × W × 3
3: # bounding_boxes : B × 4
4: # affordance_labels : B × 5
5: for (images, bounding_boxes, affordance_labels) in Dataloader do
6:     slots, attention = SOLV.Spatial(images) # slots : B × K × D_slot,   attention : B × K × N
7:     slots, attention, slot_nums = SOLV.Slot_Merger(slots, attention) # slots : B × K × D_slot,
8:         #attention : B × K × N,   slot_nums : B × 1
9:     OAcE_slots = OACE_SOLV(slots) # OAcE_slots : B × K × D_slot,
10:    masks = attention.view(B, K, H//patch_size, W//patch_size)
11:    masks = interpolate(masks, size = (H, W), mode = "bilinear") # masks : B × K × H × W
12:    seg_masks = argmax(masks, dim = 1) # seg_masks : B × H × W
13:    slots_of_obj = find_slots_of_obj(bounding_boxes,   seg_masks) # seg_masks : B × H × W
14:    Ifrepresentation_type == "OAcE"
15:        representations = OAcE_slots
16:    elif representation_type == "SOLV_Spatial"
17:        representations = slots
18:    inputs = []
19:    labels = []
20:    for i in range(B) do
21:        non_interacting_selected = False
22:        for j in range(slot_nums[i]) do
23:            If slots_of_obj == j :
24:                inputs.append(representations[i][j])
25:                labels.append(affordance_labels[i])
26:            Else:
27:                If non_interacting_selected == False:
28:                    inputs.append(representations[i][j])
29:                    labels.append([0, 0, 0, 0, 0])
30:                    non_interacting_selected = True
31:        end for
32:    end for
33:    predictions = ACM(inputs)
34:    loss = criterion(predictions, labels)
35:    loss.backward()
36:    optimizer.step()
37: end for
```

The training algorithm begins by processing the images of the dataset through the SOLV module to obtain the slot representations and the corresponding attention maps. These are then passed through the *Slot Merger Module* which combines some of the slots based on their similarity, as described previously. This evaluation aims to compare the slot representations of the *Spatial Binder* of *SOLV* to the representations of the $OAcE_{SOLV}$ (line 9). Lines 14-17 specify which representations are chosen for each training.

The attention masks are interpolated to match the images' original shape and size and are

then used to generate segmentation masks by assigning each pixel to the slot that attends to it the most (lines 10-12). For each image, the function *find_slots_of_obj*() finds the slot that claimed the most pixels within the bounding box of the interacting object (line 13). The vectors of these identified slots are then associated with the multi-label affordance targets from the *SA-Vb* dataset, while one randomly selected slot corresponding to a non-interacting object in each image is assigned a zero vector to avoid data skew (lines 20-32). Subsequently, the *ACM* is trained using these input-target pairs (lines 33-36).

Training *ACM* involves training for 20 epochs, using batches of size 18 and a learning rate scheduling approach that involves an initial linear warm-up until $lr = 0.001$, followed by exponential decay. The Smooth L1 Loss function is again used to improve robustness. Quantitative results are presented in Table 23 and qualitative results in Figures 44 and 45. In the qualitative results, the slot segmentation masks are visualized using different colors and the affordance categorization labels are positioned at the center of mass of each slot's segmentation mask.

Configuration	Recall	Precision	F1 Score	Accuracy
GT	0.7570	0.9407	0.8378	0.8793
$OAcE_{SOLV}$	**0.7109**	0.9470	**0.8103**	**0.8631**
SOLV Spatial	0.7065	0.9476	0.8076	0.8614

Table 23: SOLV representations comparison on the $SA - Vb$ dataset

5.3.2 Model Ablations

Slot Binding Iterations. In this ablation experiment (Figure 43) we test how the slot binding iterations of the *Spatial Binder* module affect the extracted representations. The results indicate that increasing the number of slot binding iterations consistently leads to improved performance across all configurations. However, as discussed above, the increase in slot binding iterations increases the complexity of the algorithm.

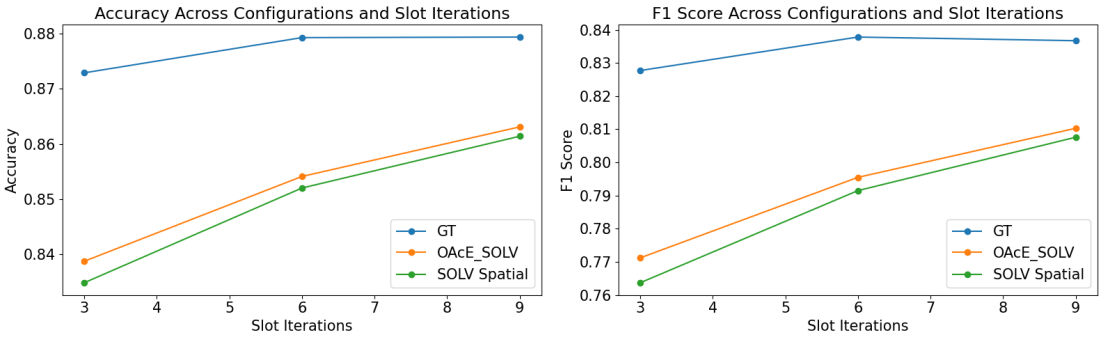


Figure 43: Accuracy and F1 score Across Configurations and Slot Iterations

5.4 Observations

To begin with, the slot representations present comparable results to the representations of the previous chapter. This is despite being smaller in size ($D_{OAcE_{SOLV}} = 128$, $D_{OAcE} = 384$) and achieving automatic segmentation, which introduces some noise into the process.

The ground truth (GT) representations come from the *Temporal Binder* module SOLV. It provides access to the encoding of a "perfect memory" involving a video clip with the interacting object. Similar to the SA-vb testing of previous chapter, the results show that these representations might contain some useful signal not present in the representations from the *Spatial Binder*. The $OAcE_{\text{SOLV}}$ tries to capture some of that useful signal, and while the results hint that it might achieve this, the improvement appears to be minimal.

An important observation is that the precision metric does not improve in the GT and $OAcE_{\text{SOLV}}$ representations. This is likely because the dataset contains significantly more negative labels than positive ones, making it preferable for the models to be conservative with their predictions. As a result, the F1 score becomes a more insightful metric, as it balances both precision and recall, offering a clearer picture of the model's performance.

Finally, the qualitative results show that even though the training process involved one object per scene, the object-centric paradigm allows the model to detect and categorize correctly multiple objects in the scene. Additionally, in some cases, a single object may be attended to by multiple slots, leading to a finer segmentation where the object is divided into several parts rather than being treated as a whole. This increased segmentation granularity can be desirable in certain scenarios but not in others. In most cases, all the slots corresponding to the same object are correctly categorized, ensuring that the object is recognized as a coherent entity despite being segmented into smaller parts.

5.5 Limitations and future directions

Dataset Limitations. Similarly to the previous chapter, a more comprehensive evaluation would involve the use of larger datasets such as Ego4D [36] and EPIC-Kitchens [17]. Additionally, it would be interesting to explore the incorporation of these representations in architectures that aim to tackle tasks like Action Anticipation [127].

Representations for control Good performance on the Affordance Categorization task can be very beneficial in the field of robot planning. Ideally, the same representation should also be useful for robots for their control tasks. However, a representation that performs well in a recognition task does not necessarily perform well in control tasks [78]. In the next chapter, we test the SOLV representations using a simple control benchmark.

5.6 Conclusion

In this experimental section, we studied different models that utilize the Slot Attention architecture and tested the slot representations of the SOLV model in the affordance categorization task using the *SA – Vb* dataset. The SOLV model is a compelling candidate for central themes of this thesis because its modular design allows for the extraction of object-centric representations from both image and video data.

By utilizing its Spatial and Temporal Binder modules, the SOLV model demonstrates the ability to process entire scenes and video clips and automatically segment them, providing object-centric representations that can then be used in downstream tasks. We show that the video representations from the *Temporal Binder* have a small advantage in the affordance categorization task compared to the static image *Spatial Binder* representations. We experimented with a variant of the Object Action-centric encoder, $OAcE_{\text{SOLV}}$, which attempts to link some of the *Temporal Binder* information to the *Spatial Binder* representations. The $OAcE_{\text{SOLV}}$ representation achieves a modest increase. Furthermore, the model's ability to detect and categorize multiple

objects in a scene, despite being trained with one object per scene, highlights its potential for generalization.

Figure 44: Qualitative examples of the *ACM* on the $OAcE_{SOLV}$ representations, Part 1

Figure 45: Qualitative examples of the *ACM* on the $OAcE_{SOLV}$ representations, Part 2

6 Slot Attention Representations for Control

6.1 Introduction

Visual pre-training methods should provide representations that can be utilized in a variety of downstream tasks and require minimal retraining [95, 70]. The task that was studied in the previous sections falls in the category of recognition tasks. However, representations that perform well in a recognition tasks do not necessarily perform well in control tasks [78]. In this chapter, we test the SOLV representations using a simple control benchmark.

We introduce a method to combine the spatial slot representations of the SOLV model to generate image representations for a simulated robot manipulation task. Specifically, this method is applied to a simulated pouring task (Figure 46) provided by the Train Offline, Test Online (TOTO) benchmark [128]. We evaluate the performance of this SOLV-based image encoder against other pre-trained image encoders that were trained on out-of-domain data. Our results demonstrate that SOLV generally achieves better performance in this setting, even in a task where an object-centric paradigm might not initially appear to provide an additional advantage. The simulated task, despite its limitations, offers a valuable first step in testing and refining our method before transitioning to real-world evaluations.

However, these results are on a simulated environment that may not transfer to real-world scenarios, which is a common issue in robotics research [128].

Figure 46: The TOTO [128] simulated robotic pouring task.

6.2 Theoretical Background

6.2.1 Reinforcement Learning

According to Russell and Norvig [93], an agent is an entity that interacts within an external environment trying to achieve an objective. Reinforcement Learning (RL) algorithms aim to develop agents that interact with an external environment in a way that maximizes the expected reward signal received from this environment [104]. This interaction is typically modeled using the Markov Decision Process framework (MDPs): The agent interacts with the environment across a series of discrete time steps, as shown in Figure 47.

During each time-step, the agent receives a signal containing information about the environment's state, $s_t \in S$, and selects an action, $a_t \in A$. This action, in turn, influences the transition to a new state, $s_{t+1} \in S$ and the reward, $r_{t+1} \in R$, received from the environment. The state transitions are governed by the dynamics of the system, which are represented by the transition probabilities denoted as $P : (S \times A)^2 \to [0, 1]$. The transition probabilities $P(s', r|s, a)$,

Notation	Description
S	State space
A	Action space
$s_t \in S$	State at time step t
$a_t \in A$	Action at time step t
$r_t \in R$	Reward at time step t
τ	An ordered list of state-action pairs: $\tau = [(s_1, a_1), (s_2, a_2), \ldots, (s_n, a_n)]$
$P(s', r\|s, a)$	Transition probability: the probability of getting reward r and transitioning to state s' from state s when action a is taken.
$\pi(a\|s)$	Stochastic policy: the probability of choosing action a when s is the current state.
$V(s)$	State Value function: the expected future return when the agent is initially at state s
$Q(s, a)$	Action-State Value function: the expected future return when the agent is initially at state s and action a is taken.

Table 24: Reinforcement Learning and Imitation Learning Notations

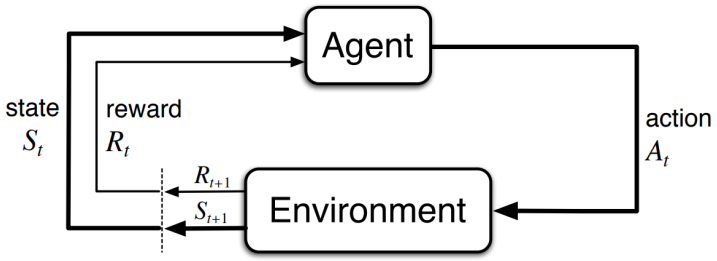


Figure 47: The the Markov Decision Process framework. Source: [104]

are connected with the uncertainty of receiving the reward r and transitioning to the state s' from the state s when action a is taken. The agent's strategy for selecting actions is expressed through the policy function $\pi : S \rightarrow A$ that maps states to actions [104].

RL can be categorized into several branches, each with unique techniques and applications (Figure **??**) [104, 1]. The primary distinction is between model-based and model-free methods. Model-based RL involves learning a model of the environment's dynamics on which the policy is developed. Model-free RL relies on trial and error to learn policies directly without direct modeling of the environment. Further distinctions include value-based and policy-based methods. Value-based approaches aim to estimate the value of states or state-action pairs, as quantified by the State Value function, $V(s)$ or the Action-State Value function, $Q(s, a)$. These algorithms utilize these functions to develop optimal policies, as in the case of Q-learning. Policy-based methods, on the other hand, directly optimize the policy, often using gradient ascent/decent techniques, as in policy gradient methods. In addition, hybrid approaches, such as actor-critic methods, combine elements of both value and policy-based methods to combine their respective strengths. RL can also be classified into on-policy and off-policy methods. On-policy methods learn policies based on actions taken by the current policy, while off-policy methods can learn from actions taken by different policies, offering greater flexibility and data efficiency.

6.2.2 Imitation Learning

Although reinforcement learning focuses on learning through interaction with the environment, imitation learning (IL) takes a different approach by learning from expert demonstrations. In IL, the agent attempts to replicate the actions of an expert (also known as a teacher in the literature), avoiding the need for extensive exploration during learning. This can be particularly beneficial in scenarios where exploration is dangerous or expensive, such as in autonomous driving or robotic manipulation [29]. The following paragraph presents the mathematical notation of IL, as described in [29] and [104].

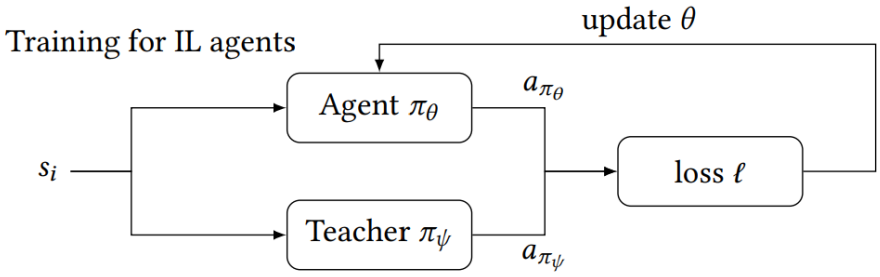


Figure 48: Training of imitation learning agents. Source: [29]

A **policy** is a function π that maps states to actions:

$$\pi : S \to A \tag{77}$$

where S is the set of possible states, and A is the set of possible actions. A policy has internal parameters ∂, which represent the variables adjusted during learning. The set of all possible policies is defined as $\Pi = \{\pi_1, \pi_2, \ldots, \pi_z\}$. An agent π_∂ is a specific realization of Π. Given an environment state $s \in S$, the agent selects an action $a \in A$ according to its learned parameters ∂. An expert π_ψ is a special case of an agent, with parameters ψ, whose behavior the IL algorithm aims to approximate.

In most cases, the IL algorithm cannot directly access the expert's policy π_ψ, because it requires knowledge of the teacher's internal state. For example, when the expert is a human, accessing their internal state is impossible. Therefore, IL uses demonstrations to learn a policy π_∂ that imitates the teacher's policy π_ψ.

A **trajectory** is an ordered sequence of state-action pairs:

$$\tau = [(s_1, a_1), (s_2, a_2), \ldots, (s_n, a_n)], \tag{78}$$

where $n > 1$, and the agent transitions to state s_{t+1} from state s_t after taking action a_t for all $t = 1, 2, \ldots, n-1$.

A **demonstration**, $d \in (S, A)$, is a state-action pair taken from a teacher's trajectory.

Behavior Cloning. (BC) is one of the earliest and simplest imitation learning (IL) algorithms [29, 9]. It applies supervised learning techniques to train a policy π_∂ that predicts the most likely action given a state, i.e., $\arg\max P(a|s)$, based on a dataset generated by the expert. More formally, given a training set $\mathcal{L}$ generated by the expert, which may consist of trajectories (ordered

sequences) $\mathcal{T} = \tau_1, \tau_2, \ldots, \tau_k$ or a set of unordered demonstrations $D = [(s_1, a_1), \ldots, (s_n, a_n)]$, the goal of the agent's policy π_ϑ is to mimic the expert's policy π_ψ. The BC process can be formulated as minimizing the loss function ℓ, computed on the expert's actions $\pi_\psi(s)$ and the agent's actions $\pi_\vartheta(s)$, as follows:

$$\arg\min_{\vartheta} \sum_{\tau \in \mathcal{L}} \sum_{s \in \tau} \ell(\pi_\psi(s), \pi_\vartheta(s)). \tag{79}$$

One disadvantage of Behavior Cloning (BC) is that in complex problems, BC-trained policies often struggle to generalize effectively. This is because the learned policy π_ϑ tends to perform poorly when encountering states not present in the expert's demonstrations, and in larger state spaces, it is unlikely that the demonstrations will uniformly cover all possible states. This difference in state distributions between expert demonstrations and the actions of the learned policy leads to compounding errors. However, BC is usually used effectively to bootstrap[9] the training of an agent before applying an RL method [29, 9].

In this thesis, in Chapter 6, we apply BC in a simple robotic simulation to evaluate visual representation learning methods for manipulation.

6.3 Robotic Manipulation Task & Dataset

TOTO Benchmark. The Train Offline, Test Online (TOTO) [128] benchmark is a robotics benchmark that aims to address the lack of standardization across research centers. TOTO provided remote research teams with access to shared robotic hardware and an open-source dataset of these tasks for offline training. The TOTO dataset consists of trajectories collected through robot teleoperation, augmented with noise, and trajectories generated by BC trained agents. The benchmark focuses on two manipulation tasks: pouring and scooping. While these tasks are commonplace for humans, variations in initial conditions and the objects used make them challenging for robots, offering valuable insights into the methods being tested.

The TOTO benchmark provides a protocol to evaluate both visual representations and policy learning methods. In this thesis, we focus solely on the evaluation of visual representations, testing them on one policy-learning method: the Behavior Cloning (BC) algorithm. As discussed in a previous section, BC is a straightforward imitation learning framework. However, as shown in the results presented by TOTO [128], BC produces the best results and is therefore used to evaluate the representations.

Simulation. The TOTO software package includes a simulation environment for the pouring task and a dataset of 108 teleoperated trajectories. This simulation was used to evaluate the method in this section. While the simulation is intended for research teams to initially test their methods and is not part of the official TOTO evaluation protocol, it allows us to assess our approach in a controlled setting before transitioning to real-world testing. As mentioned earlier, simulation results might be misleading, and overfitting to the simulated environment may hinder generalization to real-world conditions. Despite these challenges, using the simulated TOTO benchmark serves as a valuable preliminary step before conducting real-world experiments.

The simulation uses the MuJoCo physics simulation engine [108]. MuJoCo is an open source physics engine designed for model-based optimization and complex dynamical systems in contact-rich settings. The simulated robot arm is a Franka Emika Panda robot arm [41] with 7 degrees of freedom. Each joint is constrained within a specified range of positions, a practice

[9] Bootstrapping in this context refers to initializing a policy using expert demonstrations before applying a reinforcement learning (RL) further improve the policy through exploration.

that simplifies the problem by shrinking the control space but is also done for safety reasons in real-life applications.

As shown in Figure 49, the robot arm is initialized holding a cup filled with 12 small spheres. The goal is to pour as many spheres as possible into another cup. The initial joint positions and location of the target cup are randomly initialized for each experiment. A manipulation is considered successful if at least one sphere is deposited into the cup and the reward is the percentage of spheres successfully deposited in the cup.

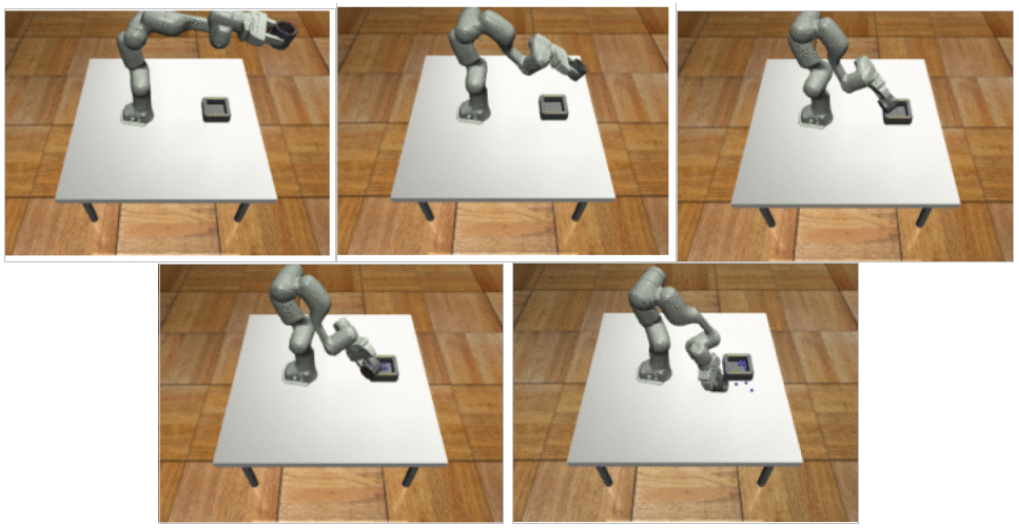


Figure 49: An example of a successful pouring sequence using the encoder proposed in this section with a reward of 75%.

The training set consists of 82 out of the 103 trajectories that are successful. Before training, all images of the training set are encoded with the visual encoder to be evaluated. Because the visual encoder is frozen during the Behavioral Cloning (BC) training, this makes the training faster, as the images would otherwise have needed to be encoded for each epoch of the policy-learning algorithm.

During training, the BC algorithm trains a neural network that acts as the agent's policy. The policy network takes as input the representation vector concatenated with the robot's current joint angles and outputs joint angle targets, which are then fed to the Mujoco controller, which moves the simulated robot arm. The input dimension is $inp\_dim = R\_dim + 7$, where R_dim is the dimension of the image representation vector. The output dimension is $out\_dim = 7 \times h$, where h is the horizon of actions to predict each time. In the following experiments $h = 10$. The architecture of the policy network is the following:

- **Normalization (Input)**:
$$\text{norm\_output} = \frac{\text{Input} - \text{inp\_mean}}{\text{inp\_std}}$$
- **Linear Layer 1**: `Linear(inp_dim, hidden_dim)`
- **ReLU Activation**: `ReLU(inplace=True)`
- **Dropout**: `nn.Dropout(p=0.1)`
- **Linear Layer 2**: `Linear(hidden_dim, hidden_dim)`
- **ReLU Activation**: `ReLU(inplace=True)`

- **Dropout**: `nn.Dropout(p=0.1)`
- **Final Linear Layer**: `Linear(hidden_dim, out_dim)`
- **Rescale Output**:

$$\text{actions} = \text{out\_mean} + \text{out\_std} \times \text{final\_layer\_output}$$

The inp_mean, inp_std, out_mean, and out_std are calculated from the training dataset and stored in buffers within the model. The input normalization step helps to fairly compare different encoders that provide inputs in various formats. Furthermore, instead of directly outputting the action vector, the network predicts how many standard deviations the output is from the action mean. This approach allows for more stable training.

6.4 Proposed Method

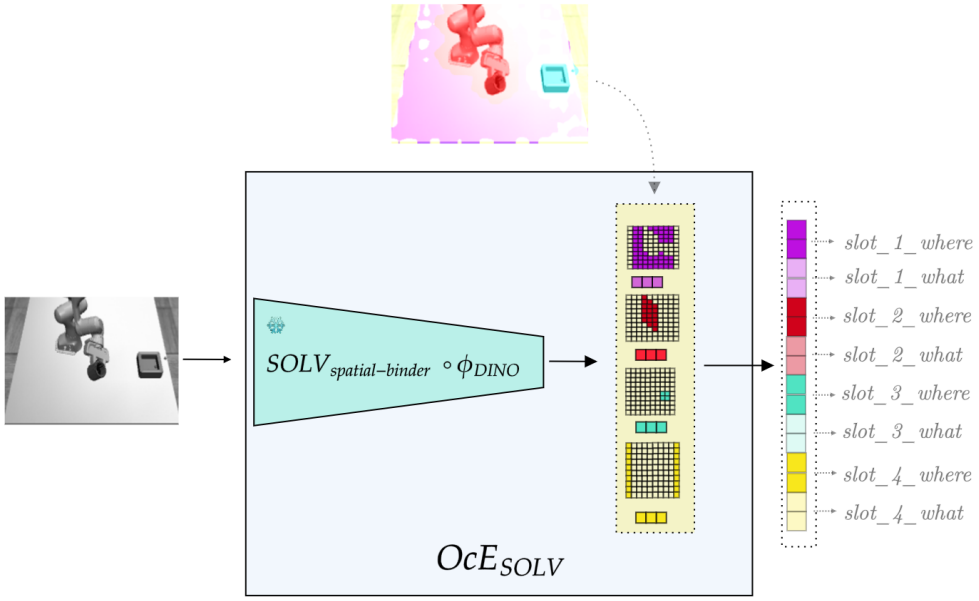


Figure 50: The $OAcE_{SOLV}$ encoder.

In this section, we introduce a method to combine the spatial slot object-centric representations of the SOLV model to generate image representations for the simulated visuomotor (Figure 51) policy training described earlier (Figure 50). The proposed is in the family of methods that use Object-Centric representations for visuomotor learning like VIOLA (Visuomotor Imitation via Object-centric LeArning) [129] and POCR (Pre-Trained Object-Centric Representations) [99].

VIOLA extracts features from object proposal regions in the input image. These features are then processed by a Transformer encoder to generate object-centric representations. The Transformer encoder allows VIOLA to capture the relationships and interactions between the detected objects.

The POCR first computes the "where" of the objects. This is achieved with the use of reference images of the robot's workspace to identify and remove background regions. Then it applies a

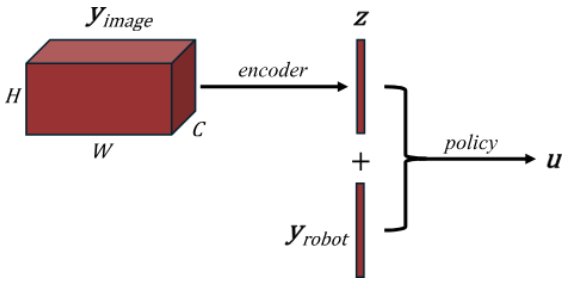


Figure 51: The visuomotor policy architecture. Source: [106].

segmentation model to the input image to extract object masks. The coordinates of these masks are the "where" part of the object-centric representation vector. Then POCR computes the "what" vector of the objects. For each object mask, POCR encodes each object using a pre-trained image encoder. By combining the "where" and "what" vectors for each slot, POCR generates object-centric representation vectors, which is then used in visuomotor policy learning.

In a similar fashion, we use the pre-trained SOLV model [3], which we fine-tuned in the previous section on the Something Something Action Videos, to produce an object-centric encoder, named $OAcE_{SOLV}$. The advantage of SOLV is that it can produce both "what" and "where" vectors at the same time through the iterative slot binding process. However, in our case, the TOTO training pipeline requires a flat vector for each image and does not include a transformer encoder capable of processing multiple object vectors. To create a flat representation vector for each image, we extract a fixed number of slots per image, generate both a "what" vector and a "where" vector for each slot, and combine them into a single representation.

As discussed in chapter 4.3, the SOLV architecture includes a Slot Merger module that merges slots based on similarity. In this section, we configure the Agglomerative Clustering algorithm to merge the initial 8 slots to 4 final slots instead of the default dynamic clustering based on a distance threshold. We also evaluate the representations resulting from different numbers of slots, and present the findings in the ablation experiments.

The four slot vectors, with a dimension of $D_{\text{what}} = 128$, produced by iterative slot binding, represent the "what" vectors. As mentioned previously, although SOLV implements Invariant Slot Attention, the slot vectors still contain some localization information due to the positional encoding of the DIVOv2 tokens.

We improve the performance of the image encoder by enriching the localization information through the attention mask of each slot. We achieve this by processing the attention mask, originally of shape $h_{att} \times w_{att} = 24 \times 36$, reducing its size to $h'_{att} \times w'_{att} = 10 \times 10$ using bilinear interpolation. Subsequently, a scaled softmax activation function is applied to the scaled down attention masks that increases the winner-takes-all competition amongst the tokens, which has been found to improve the results. The resulting "where" vectors, with a dimension of $D_{\text{where}} = 100$, are combined with the "what" vectors to produce a flat representation for the whole image of shape $D = 4 \times (128 + 100) = 912$.

6.5 Experimental Evaluation

In this section, we trained BC agents with various state-of-the-art image encoders: (i) BYOL [38], (ii) CLIP [86], (iii) DINOv2 [75], MoCo [46], Resnet50 [43]. Furthermore, we evaluated the representations of the pre-trained SOLV model provided by [3], which was trained on the Youtube-VIS 2019 real-world video dataset [118]. This evaluation aimed to determine whether

fine-tuning the SOLV model with action videos from the Something Something v2 dataset has a positive impact. The encoder based on the unfinetuned model appears in the results that follow as OcE_{SOLV} (Object-centric Encoder).

Each BC agent is trained for 80 epochs with a learning rate $lr = 0.001$. To provide a clearer picture, we train 5 BC agents on each image encoder and evaluate each one on 100 randomly initialized trajectories. The results are presented in Table 25.

	Representation size	Success Rate	Mean Reward
BYOL	512	0.46 ± 0.05	17.48 ± 2.40
CLIP	512	0.49 ± 0.06	18.61 ± 4.09
DINOv2	768	0.55 ± 0.04	18.72 ± 1.78
OcE_{SOLV_SS}	912	**0.61** ± 0.04	**25.25** ± 2.95
OcE_{SOLV_YT}	912	0.46 ± 0.06	15.07 ± 2.24
MoCo	2048	0.31 ± 0.04	9.4 ± 2.16
ResNet50	2048	0.56 ± 0.11	21.15 ± 5.60

Table 25: Comparison of Pre-trained Visual Representation Models in Training Behavior Cloning Agents for the TOTO Benchmark Simulated Pouring Task [128]. For the SOLV model, the representation size consists of 4 slots ("what") with representations of size 128, along with the corresponding scaled-down attention ("where") of size 100 each (total size 912).

6.5.1 Model Ablations

Number of slots. In Figure 52 we show the effect of configuring the AC clustering algorithm to find a specific number of slots instead of the default dynamic clustering based on a distance threshold. The results are better and more stable when the clustering algorithm is configured to find 4 slots. We hypothesize that this is due to the simplicity of environment, with only a few objects present, leading to over-segmentation when slots are 6 or 8—as seen in Figure 53.

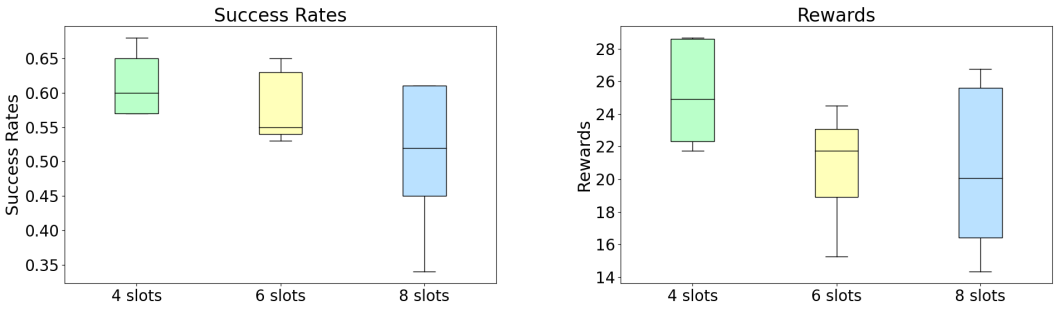
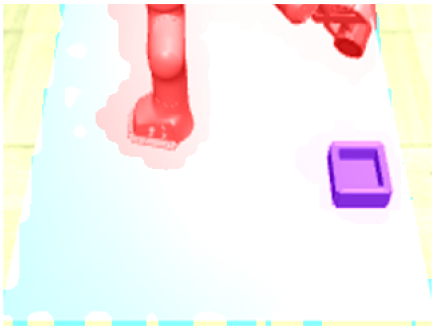
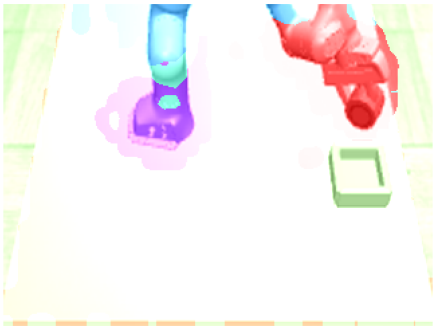


Figure 52: Mean rewards and success rates comparing different number of slots for TOTO pouring simulation. Each configuration was trained five times and evaluated accros 100 trajectories.

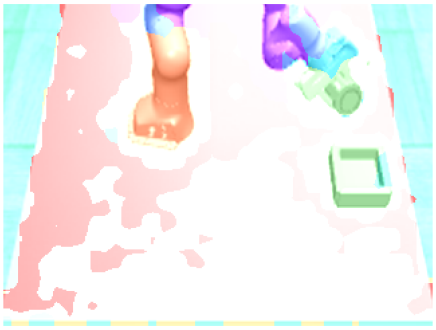
Contribution of the "where" vectors. As previously mentioned, the "what" vectors contain some localization information. Considering that this task requires the agent to have spatial awareness of itself and its environment, we tested the contribution of the "where" vectors by training BC agents using only the "what" representations. The results clearly demonstrate the



(a) 4 slots



(b) 6 slots



(c) 8 slots

Figure 53: Simulation frames from the TOTO pouring task, segmented by the Slot Attention masks of the SOLV model, with the Slot Merger module outputting 4, 6, and 8 slots.

positive contribution of the localization information derived from the processed slot attention masks, as presented in the diagrams of Figure 54.

6.6 Observations

Key observations about the above results:

- The $OAcE_{SOLV}$ module consistently produces better results, however, other encoders like DINOv2 and Resnet50 produce high-performing agents.
- The fine-tuning of the SOLV model to videos of actions has a positive impact on the representations on this control task.
- In the real-world results presented by [128], the MoCo and BYOL image encoders produced

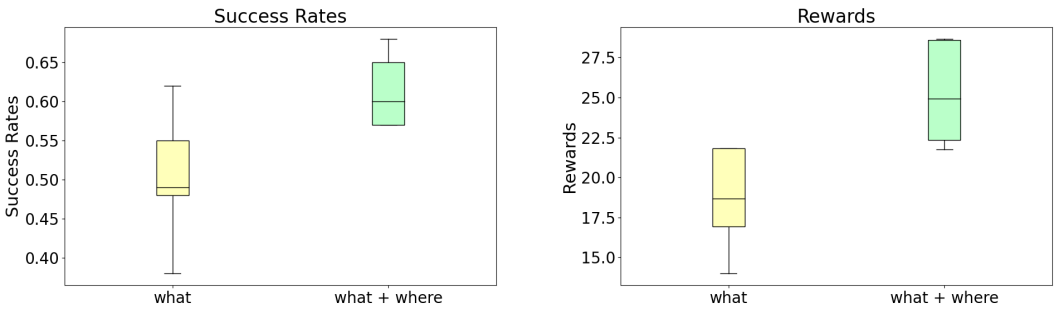


Figure 54: Mean rewards and success rates comparing "what" and complete "what" + "where" representaions. Each configuration was trained five times and evaluated across 100 trajectories.

the best performance. This highlights the inconsistency between simulation and real-world outcomes. Unfortunately, simulation results were not available for comparison.

6.7 Limitations and future directions

Evaluation. In this evaluation, we assessed the frozen image encoders by comparing their performance within a static policy learning framework. For future work, it would be interesting to expand the study by testing additional policy learning frameworks, adopting different training strategies, and performing hyperparameter tuning tailored to each image encoder. This approach could help build a better understanding of the limitations and strengths of the representation spaces provided by each representation learning method.

Simulation to Real-world. As mentioned previously, this simulated task provides a valuable first step in testing and refining this method before transitioning to real-world robot environments. In future work, it would be interesting to test this and other object-centric frameworks to explore their performance across a variety of robot manipulation and planning tasks.

Task complexity. The simulated pouring task used in this section is a simple manipulation task. It would be interesting to extend the study to more complex tasks that require planning in multi-object environments, like those provided in the Franka Kitchen [39] and Meta-world [120] environments.

Token encoder. The object-centric image encoder uses DINOv2 as the initial visual encoding step to provide dense representation vectors for the tokenized images. An interesting future research direction could involve studying an architecture similar to SOLV that extracts object-centric representations on top of visual representations specifically trained for robot manipulation. Examples include R3M [73] and LIV [69], which were trained on large egocentric human datasets, such as Ego4D [36] and EpicKitchen [17]

6.8 Conclusion

In this section, we present a method for combining the slot representations of the SOLV model to generate image representations for visuomotor learning. The proposed method is applied

to a simulated pouring task. The SOLV-based image encoder is evaluated against other out-of-domain pre-trained image encoders. The experimental results indicate that the $OAcE_{SOLV}$ module achieves positive results and that the action-centric fine-tuning process improves the representations. In conclusion, the proposed method shows promise for testing in real-world conditions.

7 Conclusion

The primary aim of this thesis was to explore methods for improving object-centric image encoders by focusing on methods that generate action-object associations based on knowledge sourced from videos of actions. These enhanced image encoders, developed through visual pre-training, are intended for use in the perception systems of robots and artificial agents.

In the first experimental section we explored a method that aims to encode action experiences, using a pre-trained Video Masked Auto-encoder, and associate them, through the use of Knowledge Distillation, with the depiction of the objects present in those experiences. We attempted to enhance two state-of-the-art pre-trained image encoders: (i) CLIP [86] and (ii) Image MAE [45]. These representations were evaluated in the task of *affordance categorization* using a small-scale dataset that we created using the Something-Something v2[35] dataset and the experiments show that the methods produce a marginal yet consistent improvement. The main disadvantage of this first method is its dependence on an object detection or segmentation module to provide the objects from a scene. As a result, in the second experimental section we focused on a model based on the Slot Attention [67] framework that automatically extracts the objects, and object-centric representations from a scene.

In the second experimental section, the object representations were drawn from the *SOLV* model [3] that achieves multi-object segmentation in real-world videos and in the process extracts representations for each of the objects in the input. The SOLV model was a compelling candidate for central themes of this thesis because its modular design allows for the extraction of object-centric representations from both image and video data. We again used the same *affordance categorization* dataset, curated from Something-Something v2[35] to evaluate are methods. The method presents competitive results while also achieving automatic segmentation of the images and a substantial reduction in per-object representation size. Furthermore, the model showcased the ability to detect and categorize multiple objects in a scene, despite being trained with one object per scene.

In the third experimental section, we tested the object-centric Slot Attention representations using a robotic control benchmark. We proposed a method to combine the spatial slot representations of the SOLV[3] model to generate image representations for visuomotor policy learning. We evaluate the performance of this image encoder against other pre-trained image encoders that were trained on out-of-domain data. Our results demonstrate that our method generally achieves better performance in this setting and the fine-tuning of the object-centric encoder using videos of actions has a positive impact on the representations. The simulated task, despite its limitations, offers a valuable first step in testing and refining our method before transitioning to real-world evaluations.

By creating action-object associations in the representations of object-centric image encoders, this study seeks to contribute to the development of more effective vision perception systems for robots and artificial agents, enabling them to better understand agent-object interaction semantics and dynamics. In future work, a more comprehensive evaluation of these methods could incorporate larger datasets like Ego4D [36] and EPIC-Kitchens [17] and further experimentation with different ways of modeling the useful content in them. Additionally, visual pre-training methods for robotics should provide representations that can be utilized in a variety of downstream tasks. As a result, it would be worthwhile to evaluate object-centric representations in a variety of manipulation, planning and recognition tasks. Finally, it would be interesting to study similar methods in the context of Continual and Lifelong Learning, and explore techniques that enable agents to encode and create associations based on their own actions and experiences.

Bibliography

[1] Joshua Achiam. “Spinning Up in Deep Reinforcement Learning”. In: (2018).

[2] Dafni Anagnostopoulou et al. “Child Engagement Estimation in Heterogeneous Child-Robot Interactions Using Spatiotemporal Visual Cues”. In: *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. 2022, pp. 3584–3589. DOI: 10.1109/IROS47612.2022.9981908.

[3] Görkay Aydemir, Weidi Xie, and Fatma Guney. “Self-supervised object-centric learning for videos”. In: *Advances in Neural Information Processing Systems* 36 (2024).

[4] Yoshua Bengio, Aaron Courville, and Pascal Vincent. “Representation learning: A review and new perspectives”. In: *IEEE transactions on pattern analysis and machine intelligence* 35.8 (2013), pp. 1798–1828.

[5] Yoshua Bengio et al. “Curriculum learning”. In: *Proceedings of the 26th annual international conference on machine learning*. 2009, pp. 41–48.

[6] Aude Billard and Danica Kragic. “Trends and challenges in robot manipulation”. In: *Science* 364.6446 (2019), eaat8414. DOI: 10.1126/science.aat8414. eprint: https://www.science.org/doi/pdf/10.1126/science.aat8414. URL: https://www.science.org/doi/abs/10.1126/science.aat8414.

[7] Christopher M Bishop and Nasser M Nasrabadi. *Pattern recognition and machine learning*. Vol. 4. 4. Springer, 2006.

[8] Ondrej Biza et al. “Invariant Slot Attention: Object Discovery with Slot-Centric Reference Frames”. In: *International Conference on Machine Learning*. PMLR. 2023, pp. 2507–2527.

[9] Jeannette Bohg. *Principles of Robot Autonomy II: Learning-based Approaches to Manipulation & Interactive Perception*. Lecture notes for CS237B. Accessed: September 3, 2024. 2023. URL: https://web.stanford.edu/class/cs237b/pdfs/lecture/lecture_10111213.pdf.

[10] Roberto Bottini and Christian F Doeller. “Knowledge across reference frames: Cognitive maps and image spaces”. In: *Trends in Cognitive Sciences* 24.8 (2020), pp. 606–619.

[11] Richard H Byrd et al. “A limited memory algorithm for bound constrained optimization”. In: *SIAM Journal on scientific computing* 16.5 (1995), pp. 1190–1208.

[12] Weilin Cai et al. “A survey on mixture of experts”. In: *arXiv preprint arXiv:2407.06204* (2024).

[13] Dongpan Chen et al. “A survey of visual affordance recognition based on deep learning”. In: *IEEE Transactions on Big Data* (2023).

[14] Hongrong Cheng, Miao Zhang, and Javen Qinfeng Shi. “A Survey on Deep Neural Network Pruning: Taxonomy, Comparison, Analysis, and Recommendations”. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* (2024), pp. 1–20. DOI: 10.1109/TPAMI.2024.3447085.

[15] Kyunghyun Cho et al. “Learning Phrase Representations using RNN Encoder–Decoder for Statistical Machine Translation”. In: *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Ed. by Alessandro Moschitti, Bo Pang, and Walter Daelemans. Doha, Qatar: Association for Computational Linguistics, Oct. 2014, pp. 1724–1734. DOI: 10.3115/v1/D14-1179. URL: https://aclanthology.org/D14-1179.

[16] Navneet Dalal and Bill Triggs. “Histograms of oriented gradients for human detection”. In: *2005 IEEE computer society conference on computer vision and pattern recognition (CVPR'05)*. Vol. 1. Ieee. 2005, pp. 886–893.

[17] Dima Damen et al. “Scaling egocentric vision: The epic-kitchens dataset”. In: *Proceedings of the European conference on computer vision (ECCV)*. 2018, pp. 720–736.

[18] Jurafsky Daniel and H Martin James. *Speech and Language Processing*. 2000.

[19] Jacob Devlin et al. “BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding”. In: 2018. URL: https://arxiv.org/abs/1810.04805.

[20] Ioanna Diamanti et al. “ViDaS Video Depth-aware Saliency Network”. In: *arXiv preprint arXiv:2305.11729* (2023).

[21] Alexey Dosovitskiy et al. “An image is worth 16x16 words: Transformers for image recognition at scale”. In: *arXiv preprint arXiv:2010.11929* (2020).

[22] Niki Efthymiou et al. “ChildBot: Multi-robot perception and interaction with children”. In: *Robotics and Autonomous Systems* 150 (2022), p. 103975. ISSN: 0921-8890. DOI: https://doi.org/10.1016/j.robot.2021.103975. URL: https://www.sciencedirect.com/science/article/pii/S0921889021002426.

[23] Niki Efthymiou et al. “Visual Robotic Perception System with Incremental Learning for Child–Robot Interaction Scenarios”. In: *Technologies* 9.4 (2021). ISSN: 2227-7080. DOI: 10.3390/technologies9040086. URL: https://www.mdpi.com/2227-7080/9/4/86.

[24] Sarah FV Eiteljoerge et al. “Word-object and action-object association learning across early development”. In: *Plos one* 14.8 (2019), e0220317.

[25] Georgios Evangelopoulos et al. “Multimodal saliency and fusion for movie summarization based on aural, visual, and textual attention”. In: *IEEE Transactions on Multimedia* 15.7 (2013), pp. 1553–1568.

[26] Zhiyuan Fang et al. “Seed: Self-supervised distillation for visual representation”. In: *arXiv preprint arXiv:2101.04731* (2021).

[27] Christoph Feichtenhofer et al. “Slowfast networks for video recognition”. In: *Proceedings of the IEEE/CVF international conference on computer vision*. 2019, pp. 6202–6211.

[28] Peng Gao et al. “Mimic before reconstruct: Enhancing masked autoencoders with feature mimicking”. In: *International Journal of Computer Vision* 132.5 (2024), pp. 1546–1556.

[29] Nathan Gavenski et al. “A Survey of Imitation Learning Methods, Environments and Metrics”. In: (2024).

[30] James J Gibson. *The ecological approach to visual perception*. Psychology press, 1986.

[31] Athanasios Glentis Georgoulakis, George Retsinas, and Petros Maragos. “Feather: An Elegant Solution to Effective DNN Sparsification”. In: *Proceedings of the British Machine Vision Conference (BMVC)*. 2023.

[32] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. http://www.deeplearningbook.org. MIT Press, 2016.

[33] Marco Gori, Alessandro Betti, and Stefano Melacci. *Machine Learning: A constraint-based approach*. Elsevier, 2023.

[34] Jianping Gou et al. “Knowledge distillation: A survey”. In: *International Journal of Computer Vision* 129.6 (2021), pp. 1789–1819.

[35] Raghav Goyal et al. “The" something something" video database for learning and evaluating visual common sense”. In: *Proceedings of the IEEE international conference on computer vision*. 2017, pp. 5842–5850.

[36] Kristen Grauman et al. “Ego4D: Around the World in 3,000 Hours of Egocentric Video”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. June 2022, pp. 18995–19012.

[37] Samuel Greydanus et al. “Visualizing and understanding atari agents”. In: *International conference on machine learning*. PMLR. 2018, pp. 1792–1801.

[38] Jean-Bastien Grill et al. “Bootstrap your own latent-a new approach to self-supervised learning”. In: *Advances in neural information processing systems* 33 (2020), pp. 21271–21284.

[39] Abhishek Gupta et al. “Relay policy learning: Solving long-horizon tasks via imitation and reinforcement learning”. In: *arXiv preprint arXiv:1910.11956* (2019).

[40] Rajendra P Gupta. “JWST early Universe observations and ΛCDM cosmology”. In: *Monthly Notices of the Royal Astronomical Society* 524.3 (2023), pp. 3385–3395.

[41] Sami Haddadin et al. “The franka emika robot: A reference platform for robotics research and education”. In: *IEEE Robotics & Automation Magazine* 29.2 (2022), pp. 46–64.

[42] Mohammed Hassanin, Salman Khan, and Murat Tahtali. “Visual affordance and function understanding: A survey”. In: *ACM Computing Surveys (CSUR)* 54.3 (2021), pp. 1–35.

[43] Kaiming He et al. “Deep residual learning for image recognition”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016, pp. 770–778.

[44] Kaiming He et al. “Mask r-cnn”. In: *Proceedings of the IEEE international conference on computer vision*. 2017, pp. 2961–2969.

[45] Kaiming He et al. “Masked autoencoders are scalable vision learners”. In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2022, pp. 16000–16009.

[46] Kaiming He et al. “Momentum contrast for unsupervised visual representation learning”. In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2020, pp. 9729–9738.

[47] Alexandre Heuillet, Fabien Couthouis, and Natalia Díaz-Rodríguez. “Explainability in deep reinforcement learning”. In: *Knowledge-Based Systems* 214 (2021), p. 106685. ISSN: 0950-7051. DOI: https://doi.org/10.1016/j.knosys.2020.106685. URL: https://www.sciencedirect.com/science/article/pii/S0950705120308145.

[48] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. “Distilling the knowledge in a neural network”. In: *arXiv preprint arXiv:1503.02531* (2015).

[49] Eleni Ilkou and Maria Koutraki. “Symbolic vs sub-symbolic ai methods: Friends or enemies?” In: *CIKM (Workshops)*. Vol. 2699. 2020.

[50] Sergey Ioffe. “Batch normalization: Accelerating deep network training by reducing internal covariate shift”. In: *arXiv preprint arXiv:1502.03167* (2015).

[51] Alexandros Iosifidis and Anastasios Tefas. *Deep learning for robot perception and cognition*. Academic Press, 2022.

[52] Ya Jing et al. “Exploring Visual Pre-training for Robot Manipulation: Datasets, Models and Methods”. In: *2023 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. 2023, pp. 11390–11395. DOI: 10.1109/IROS55552.2023.10342201.

[53] Sheikh Musa Kaleem et al. “A Comprehensive Review of Knowledge Distillation in Computer Vision”. In: *arXiv preprint arXiv:2404.00936* (2024).

[54] Siddharth Karamcheti et al. “Language-driven representation learning for robotics”. In: *arXiv preprint arXiv:2302.12766* (2023).

[55] Zeyad Khalifa and Syed Afaq Ali Shah. “A Large Scale Multi-View RGBD Visual Affordance Learning Dataset”. In: *2023 IEEE International Conference on Image Processing (ICIP)*. IEEE. 2023, pp. 1325–1329.

[56] Diederik P Kingma. “Adam: A method for stochastic optimization”. In: *arXiv preprint arXiv:1412.6980* (2014).

[57] Alexander Kirillov et al. “Segment anything”. In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 2023, pp. 4015–4026.

[58] Petros Koutras and Petros Maragos. “A perceptually based spatio-temporal computational framework for visual saliency estimation”. In: *Signal Processing: Image Communication* 38 (2015), pp. 15–31.

[59] Petros Koutras and Petros Maragos. “SUSiNet: See, Understand and Summarize it”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*. Jan. 1, 2019. URL: http://robotics.ntua.gr/wp-content/uploads/sites/2/Koutras_SUSiNet_See_Understand_and_Summarize_It_CVPRW_2019_paper.pdf. published.

[60] Petros Koutras, Athanasia Zlatinsi, and Petros Maragos. “Exploring CNN-Based Architectures for Multimodal Salient Event Detection in Videos”. In: *2018 IEEE 13th Image, Video, and Multidimensional Signal Processing Workshop (IVMSP)*. 2018, pp. 1–5. DOI: 10.1109/IVMSPW.2018.8448977.

[61] Jitendra Malik Kristen Grauman et al., eds. *CVPR 2023 Workshop*. IEEE. 2023. URL: https://www.youtube.com/watch?v=zGMT74GPn0U.

[62] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. “Deep learning”. In: *nature* 521.7553 (2015), pp. 436–444.

[63] Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. “Layer normalization”. In: *ArXiv e-prints* (2016).

[64] Sergey Levine et al. “End-to-end training of deep visuomotor policies”. In: *Journal of Machine Learning Research* 17.39 (2016), pp. 1–40.

[65] Ruyang Liu et al. “Revisiting temporal modeling for clip-based image-to-video knowledge transferring”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2023, pp. 6555–6564.

[66] Xiaowei Liu, Yikun Hu, and Jianguo Chen. “Hybrid CNN-Transformer model for medical image segmentation with pyramid convolution and multi-layer perceptron”. In: *Biomedical Signal Processing and Control* 86 (2023), p. 105331.

[67] Francesco Locatello et al. “Object-centric learning with slot attention”. In: *Advances in neural information processing systems* 33 (2020), pp. 11525–11538.

[68] David G Lowe. “Object recognition from local scale-invariant features”. In: *Proceedings of the seventh IEEE international conference on computer vision*. Vol. 2. Ieee. 1999, pp. 1150–1157.

[69] Yecheng Jason Ma et al. “Liv: Language-image representations and rewards for robotic control”. In: *International Conference on Machine Learning*. PMLR. 2023, pp. 23301–23320.

[70] Jitendra Malik. *Visual Pre-training for Robotics*. YouTube video. CVPR 2023 Workshop. 2023. URL: https://www.youtube.com/watch?v=zGMT74GPn0U.

[71] Ričards Marcinkevičs and Julia E Vogt. "Interpretability and explainability: A machine learning zoo mini-tour". In: *arXiv preprint arXiv:2012.01805* (2020).

[72] *Something-Else: Compositional Action Recognition with Spatial-Temporal Interaction Networks*. 2020.

[73] Suraj Nair et al. "R3m: A universal visual representation for robot manipulation". In: *arXiv preprint arXiv:2203.12601* (2022).

[74] Niall O'Mahony et al. "Deep learning vs. traditional computer vision". In: *Advances in Computer Vision: Proceedings of the 2019 Computer Vision Conference (CVC), Volume 1 1*. Springer. 2020, pp. 128–144.

[75] Maxime Oquab et al. "Dinov2: Learning robust visual features without supervision". In: *arXiv preprint arXiv:2304.07193* (2023).

[76] François Osiurak, Yves Rossetti, and Arnaud Badets. "What is an affordance? 40 years later". In: *Neuroscience & Biobehavioral Reviews* 77 (2017), pp. 403–417.

[77] Stephen E Palmer. *Vision science: Photons to phenomenology*. MIT press, 1999.

[78] Simone Parisi et al. "The unsurprising effectiveness of pre-trained vision models for control". In: *international conference on machine learning*. PMLR. 2022, pp. 17359–17371.

[79] Wonpyo Park et al. "Relational knowledge distillation". In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2019, pp. 3967–3976.

[80] Nikhil Parthasarathy et al. "Self-supervised video pretraining yields robust and more human-aligned visual representations". In: *Proceedings of the 37th International Conference on Neural Information Processing Systems*. NIPS '23. New Orleans, LA, USA: Curran Associates Inc., 2024.

[81] Adam Paszke et al. "PyTorch: An Imperative Style, High-Performance Deep Learning Library". In: *Advances in Neural Information Processing Systems 32*. Curran Associates, Inc., 2019, pp. 8024–8035. URL: http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf.

[82] Fabian Pedregosa et al. "Scikit-learn: Machine learning in Python". In: *the Journal of machine Learning research* 12 (2011), pp. 2825–2830.

[83] Chiara Plizzari et al. "An Outlook into the Future of Egocentric Vision". In: *International Journal of Computer Vision* (May 2024). ISSN: 1573-1405. DOI: 10.1007/s11263-024-02095-7. URL: https://doi.org/10.1007/s11263-024-02095-7.

[84] Thomas Porter and Tom Duff. "Compositing digital images". In: *Proceedings of the 11th annual conference on Computer graphics and interactive techniques*. 1984, pp. 253–259.

[85] PyTorch. *CrossEntropyLoss*. [Accessed: Aug. 16, 2024]. 2024. URL: https://pytorch.org/docs/stable/generated/torch.nn.CrossEntropyLoss.html.

[86] Alec Radford et al. "Learning transferable visual models from natural language supervision". In: *International conference on machine learning*. PMLR. 2021, pp. 8748–8763.

[87] Ilija Radosavovic et al. "Real-World Robot Learning with Masked Visual Pre-training". In: *CoRL* (2022).

[88] Ilija Radosavovic et al. "Real-world robot learning with masked visual pre-training". In: *Conference on Robot Learning*. PMLR. 2023, pp. 416–426.

[89] Joseph Redmon et al. “You only look once: Unified, real-time object detection”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016, pp. 779–788.

[90] George Retsinas et al. “3D Facial Expressions through Analysis-by-Neural-Synthesis”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. June 2024, pp. 2490–2501.

[91] George Retsinas et al. “Online Weight Pruning Via Adaptive Sparsity Loss”. In: *2021 IEEE International Conference on Image Processing (ICIP)*. 2021, pp. 3517–3521. DOI: 10.1109/ICIP42928.2021.9506301.

[92] Rob Romijnders et al. “Representation learning from videos in-the-wild: An object-centric approach”. In: *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*. 2021, pp. 177–187.

[93] Stuart J Russell and Peter Norvig. *Artificial intelligence: a modern approach*. Pearson, 2016.

[94] Erol Şahin et al. “To afford or not to afford: A new formalization of affordances toward affordance-based robot control”. In: *Adaptive Behavior* 15.4 (2007), pp. 447–472.

[95] Alexander Sax et al. “Learning to Navigate Using Mid-Level Visual Priors”. In: *Proceedings of the Conference on Robot Learning*. Ed. by Leslie Pack Kaelbling, Danica Kragic, and Komei Sugiura. Vol. 100. Proceedings of Machine Learning Research. PMLR, Nov. 2020, pp. 791–812. URL: https://proceedings.mlr.press/v100/sax20a.html.

[96] scikit-learn. *Clustering*. https://scikit-learn.org/stable/modules/clustering.html. Accessed: Aug. 12, 2024.

[97] *scikit-learn: machine learning in Python — scikit-learn 1.5.2 documentation*. https://scikit-learn.org. Accessed: [Insert date here].

[98] Shai Shalev-Shwartz and Shai Ben-David. *Understanding machine learning: From theory to algorithms*. Cambridge university press, 2014.

[99] Junyao Shi* et al. “Composing Pre-Trained Object-Centric Representations for Robotics From" What" and" Where" Foundation Models”. In: *ICRA* (2024).

[100] Petru Soviany et al. “Curriculum learning: A survey”. In: *International Journal of Computer Vision* 130.6 (2022), pp. 1526–1565.

[101] Nitish Srivastava et al. “Dropout: a simple way to prevent neural networks from overfitting”. In: *The journal of machine learning research* 15.1 (2014), pp. 1929–1958.

[102] Supervisely. *Supervisely Computer Vision platform*. https://supervisely.com. Computer Vision Tools. visited on 2023-07-20. July 2023. URL: https://supervisely.com.

[103] Richard Sutton. “The bitter lesson”. In: *Incomplete Ideas (blog)* 13.1 (2019), p. 38.

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

[105] PyTorch Team. *PyTorch Documentation: Data Loading*. Accessed: [Current Date]. 2023. URL: https://pytorch.org/docs/stable/data.html.

[106] Russ Tedrake. *Underactuated Robotics. Algorithms for Walking, Running, Swimming, Flying, and Manipulation*. 2023. URL: https://underactuated.csail.mit.edu.

[107] Sergios Theodoridis. *Machine learning: a Bayesian and optimization perspective*. Academic Press, 2020.

[108] Emanuel Todorov, Tom Erez, and Yuval Tassa. “Mujoco: A physics engine for model-based control”. In: *2012 IEEE/RSJ international conference on intelligent robots and systems*. IEEE. 2012, pp. 5026–5033.

[109] Zhan Tong et al. “Videomae: Masked autoencoders are data-efficient learners for self-supervised video pre-training”. In: *Advances in neural information processing systems* 35 (2022), pp. 10078–10093.

[110] Antigoni Tsiami, Petros Koutras, and Petros Maragos. “Stavis: Spatio-temporal audiovisual saliency network”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2020, pp. 4766–4776.

[111] Ashish Vaswani et al. “Attention is all you need”. In: *Advances in neural information processing systems* 30 (2017).

[112] Rui Wang et al. “Masked video distillation: Rethinking masked feature modeling for self-supervised video representation learning”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2023.

[113] Nicholas Watters et al. “Spatial broadcast decoder: A simple architecture for learning disentangled representations in vaes”. In: *arXiv preprint arXiv:1901.07017* (2019).

[114] Thaddäus Wiedemer et al. “Provable Compositional Generalization for Object-Centric Learning”. In: *arXiv preprint arXiv:2310.05327* (2023).

[115] Ian H Witten et al. *Data mining: practical machine learning tools and techniques*. 2017.

[116] Mingle Xu et al. “A comprehensive survey of image augmentation techniques for deep learning”. In: *Pattern Recognition* 137 (2023), p. 109347.

[117] Natsuki Yamanobe et al. “A brief review of affordance in robotic manipulation research”. In: *Advanced Robotics* 31.19-20 (2017), pp. 1086–1101.

[118] Linjie Yang, Yuchen Fan, and Ning Xu. “Video instance segmentation”. In: *Proceedings of the IEEE/CVF international conference on computer vision*. 2019, pp. 5188–5197.

[119] Wei Yi and S Marshall. “Principal component analysis in application to object orientation”. In: *Geo-spatial Information Science* 3.3 (2000), pp. 76–78.

[120] Tianhe Yu et al. “Meta-world: A benchmark and evaluation for multi-task and meta reinforcement learning”. In: *Conference on robot learning*. PMLR. 2020, pp. 1094–1100.

[121] Chaoning Zhang et al. “A Survey on Masked Autoencoder for Visual Self-supervised Learning.” In: *IJCAI*. 2023, pp. 6805–6813.

[122] Chi Zhang, Karthika Mohan, and Judea Pearl. “Causal Inference under Interference and Model Uncertainty”. In: *Conference on Causal Learning and Reasoning*. PMLR. 2023, pp. 371–385.

[123] *Fine-grained egocentric hand-object segmentation: Dataset, model, and applications*. Springer. 2022, pp. 127–145.

[124] Yuhao Zhang et al. “Contrastive learning of medical visual representations from paired images and text”. In: *Machine Learning for Healthcare Conference*. PMLR. 2022, pp. 2–25.

[125] Zhigen Zhao et al. *A Survey of Optimization-based Task and Motion Planning: From Classical To Learning Approaches*. June 2024.

[126] Minghang Zheng et al. “End-to-end object detection with adaptive clustering transformer”. In: *arXiv preprint arXiv:2011.09315* (2020).

[127] Zeyun Zhong et al. “A survey on deep learning techniques for action anticipation”. In: *arXiv preprint arXiv:2309.17257* (2023).

[128] Gaoyue Zhou et al. “Train offline, test online: A real robot learning benchmark”. In: *2023 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2023, pp. 9197–9203.

[129] Yifeng Zhu et al. “Viola: Imitation learning for vision-based manipulation with object proposal priors”. In: *Conference on Robot Learning*. PMLR. 2023, pp. 1199–1210.

[130] Eric R Ziegel. *The elements of statistical learning*. 2003.

[131] A Zlatintsi et al. “COGNIMUSE: a multimodal video database annotated with saliency, events, semantics and emotion with application to summarization”. In: *EURASIP Journal on Image and Video Processing* 54 (Jan. 1, 2017), pp. 1–24. DOI: doi10.1186/s13640-017-0194. URL: http://robotics.ntua.gr/wp-content/publications/Zlatintsi+_COGNIMUSEdb_EURASIP_JIVP-2017.pdf. published.