Εθνικο Μετσοβιο Πολυτεχνειο
Σχολη Ηλεκτρολογων Μηχανικων και Μηχανικων Υπολογιστων
Τομεας Τεχνολογιας Πληροφορικης και Υπολογιστων
Εργαστηριο Συστηματων Τεχνητης Νοημοσυνης και Μαθησης

Analysis of modern domain generalization methods in autonomous driving using data augmentation and contrastive learning

Διπλωματικη Εργασια

του

ΣΥΡΙΓΩΝΑΚΗ ΑΛΕΞΑΝΔΡΟΥ

Επιβλέπων: Αθανάσιος Βουλόδημος
Επίκουρος Καθηγητής Ε.Μ.Π.

Αθήνα, Ιούλιος 2025

Εθνικο Μετσοβιο Πολυτεχνειο
Σχολη Ηλεκτρολογων Μηχανικων και Μηχανικων Υπολογιστων
Τομεας Τεχνολογιας Πληροφορικης και Υπολογιστων
Εργαστηριο Συστηματων Τεχνητης Νοημοσυνης και Μαθησης

Analysis of modern domain generalization methods in autonomous driving using data augmentation and contrastive learning

Διπλωματικη Εργασια

του

ΣΥΡΙΓΩΝΑΚΗ ΑΛΕΞΑΝΔΡΟΥ

Επιβλέπων: Αθανάσιος Βουλόδημος
Επίκουρος Καθηγητής Ε.Μ.Π.

Εγκρίθηκε από την τριμελή εξεταστική επιτροπή την 4η Ιουλίου 2025.

(Υπογραφή)	*(Υπογραφή)*	*(Υπογραφή)*
.....	.....	.....
Αθανάσιος Βουλόδημος	Γιώργος Στάμου	Ανδρέας-Γεώργιος Σταφυλοπάτης
Επίκουρος Καθηγητής Ε.Μ.Π.	Καθηγητής Ε.Μ.Π.	Ομότιμος Καθηγητής Ε.Μ.Π.

Αθήνα, Ιούλιος 2025

Εθνικο Μετσοβιο Πολυτεχνειο
Σχολη Ηλεκτρολογων Μηχανικων και Μηχανικων Υπολογιστων
Τομεας Τεχνολογιας Πληροφορικης και Υπολογιστων
Εργαστηριο Συστηματων Τεχνητης Νοημοσυνης και Μαθησης

ΔΗΛΩΣΗ ΜΗ ΛΟΓΟΚΛΟΠΗΣ ΚΑΙ ΑΝΑΛΗΨΗΣ ΠΡΟΣΩΠΙΚΗΣ ΕΥΘΥΝΗΣ

Με πλήρη επίγνωση των συνεπειών του νόμου περί πνευματικών δικαιωμάτων, δηλώνω ενυπογράφως ότι είμαι αποκλειστικός συγγραφέας της παρούσας Πτυχιακής Εργασίας, για την ολοκλήρωση της οποίας κάθε βοήθεια είναι πλήρως αναγνωρισμένη και αναφέρεται λεπτομερώς στην εργασία αυτή. Έχω αναφέρει πλήρως και με σαφείς αναφορές, όλες τις πηγές χρήσης δεδομένων, απόψεων, θέσεων και προτάσεων, ιδεών και λεκτικών αναφορών, είτε κατά κυριολεξία είτε βάσει επιστημονικής παράφρασης. Αναλαμβάνω την προσωπική και ατομική ευθύνη ότι σε περίπτωση αποτυχίας στην υλοποίηση των ανωτέρω δηλωθέντων στοιχείων, είμαι υπόλογος έναντι λογοκλοπής, γεγονός που σημαίνει αποτυχία στην Πτυχιακή μου Εργασία και κατά συνέπεια αποτυχία απόκτησης του Τίτλου Σπουδών, πέραν των λοιπών συνεπειών του νόμου περί πνευματικών δικαιωμάτων. Δηλώνω, συνεπώς, ότι αυτή η Πτυχιακή Εργασία προετοιμάστηκε και ολοκληρώθηκε από εμένα προσωπικά και αποκλειστικά και ότι, αναλαμβάνω πλήρως όλες τις συνέπειες του νόμου στην περίπτωση κατά την οποία αποδειχθεί, διαχρονικά, ότι η εργασία αυτή ή τμήμα της δεν μου ανήκει διότι είναι προϊόν λογοκλοπής άλλης πνευματικής ιδιοκτησίας.

(Υπογραφή)

.....
Αλέξανδρος Συριγωνάκης
Διπλωματούχος Ηλεκτρολόγος Μηχανικός και Μηχανικός Υπολογιστών, Ε.Μ.Π.

4η Ιουλίου 2025

Περίληψη

Η ανάπτυξη νέων μοντέλων βαθιάς μάθησης έχει οδηγήσει σε σημαντική πρόοδο τον τομέα της Όρασης Υπολογιστών. Η χρήση Συνελικτικών Νευρωνικών Δικτύων σε εργασίες κατάτμησης εικόνων προσέφερε μοντέλα με πρωτοφανείς ικανότητες. Παρά την επιτυχία τους, τα μοντέλα αυτά αντιμετωπίζουν δυσκολία στη γενίκευση σε νέα δεδομένα. Κατά την εκπαίδευση, θεωρείται ότι τα δεδομένα εκπαίδευσης και ελέγχου ακολουθούν την ίδια κατανομή, κάτι που σπάνια ισχύει για πραγματικά δεδομένα. Ο τομέας της Γενίκευσης Πεδίου έχει διαμορφωθεί ως ένας νέος ερευνητικός τομέας με στόχο να βελτιωθούν οι ικανότητες γενίκευσης των μοντέλων βαθιάς μάθησης. Με την αξιοποίηση τεχνικών Γενίκευσης Πεδίου έχει επιτευχθεί η βελτίωση της απόδοσης κλασικών μοντέλων σε νέα δεδομένα, μειώνοντας την ανάγκη επανεκπαίδευσής τους. Οι τεχνικές αυτές έχουν εφαρμοστεί σε εργασίες κατάτμησης εικόνων με σημαντική επιτυχία. Η ικανότητα γενίκευσης έχει ιδιαίτερη σημασία σε περιπτώσεις που η επιβλεπόμενη μάθηση συναντά δυσκολίες, λόγω έλλειψης αξιόλογων συνόλων δεδομένων ή μεγάλων διαφορών μεταξύ των εξεταζόμενων δεδομένων, και οι περισσότερες ερευνητικές προτάσεις προσανατολίζονται στην επίλυση των προβλημάτων αυτών. Στην εργασία αυτή εξετάζουμε την ικανότητα γενίκευσης του CCSDG μοντέλου σε διαφορετικές συνθήκες, αξιοποιώντας μία FCN αρχιτεκτονική ως το βασικό μοντέλο. Επιλέχθηκε ως σύνολο δεδομένων το SYNTHIA, μία συλλογή συνθετικών εικόνων από αστικά περιβάλλοντα. Το μοντέλο εκπαιδεύτηκε και ελέγχθηκε στις περιβαλλοντολογικές και γεωγραφικές συνθήκες που παρέχονται από την έκδοση των ακολουθιών βίντεο του SYNTHIA. Το μοντέλο DiffuseMix χρησιμοποιήθηκε για να δημιουργήσει επαυξημένες εικόνες, με την εισαγωγή προτρεπτικών λέξεων. Παρατηρούμε ότι για κάποιους συνδυασμούς παραμέτρων και συνθηκών η μέθοδος γενίκευσης μπορεί να επιτύχει βελτίωση της απόδοσης. Ιδιαίτερα για το πεδίο ελέγχου που έχει την μεγαλύτερη σχέση με το πεδίο εκπαίδευσης, η μέθοδος φαίνεται να βελτιώνει την επίδοση του μοντέλου, για τις συνθήκες που διαφέρουν από την συνθήκη εκπαίδευσης. Παρόλα αυτά, δεν μπορεί να εξαχθεί κάποιο καθαρό συμπέρασμα για την καλύτερη αρχιτεκτονική, οπότε απαιτείται περαιτέρω διερεύνηση της συγκεκριμένης πρότασης.

Λέξεις Κλειδιά

Όραση Υπολογιστών, Σημασιολογική Κατάτμηση, Γενίκευση Πεδίου, Αντικρουόμενη Μάθηση, Επαύξηση Δεδομένων, Μοντέλα Διάχυσης

Abstract

The development of new Deep Learning models has led to many advancements in the field of Computer Vision. The introduction of Convolutional Neural Networks to image segmentation tasks provided models with unseen capabilities. These models, however successful, encounter obstacles when they are applied to unseen domains. In training it is assumed that the training and the testing data follow the same distribution, when in real-life settings this is rarely the case. Thus a new direction of research has been developing, with the task of improving the generalizing capabilities of deep learning models. Domain Generalization methods have been improving the out of distribution performance of basic models, expanding their capabilities without the need to train on new datasets. These methods are applied in many computer vision settings, in particular in image segmentation tasks, with substancial success. Generalization promoting techniques are crucial in tasks where supervised training confronts many difficulties, due to limited availability of useful annotated datasets or vast discrepanies between datasets, and most research approaches are aimed at overcoming these issues. In this thesis we test the generalization capabilities of the CCSDG module across different conditions, using a classic FCN architecture as the backbone. We opted to use the SYNTHIA dataset, a collection of synthetic captions of an urban environment. We trained and tested the pipeline across different environmental and geographical conditions, provided in the video sequencies version of the dataset. The DiffuseMix pipeline is also utilized to create an array of augmented images, using a textual prompt from a predetermined list. We observed that for some combinations of parameters and environments this generalization method can perform better than the baseline. In particular, for the domain most similar to the training domain, our method seem to boost the performance of the model for the conditions most dissimilar to training. However, there is no clear indication to a best performing configuration, so the capabilities of our proposal need further investigation.

Keywords

Computer Vision, Semantic Segmentation, Domain Generalization, Contrastive Learning, Data Augmentation, Diffusion Models

Ευχαριστίες

Θα ήθελα καταρχήν να ευχαριστήσω τον καθηγητή κ. Αθανάσιο Βουλόδημο για την ευκαιρία που μου έδωσε να εκπονήσω τη διπλωματική μου εργασία στο εργαστήριο Συστημάτων Τεχνητής Νοημοσύνης και Μάθησης και για την επίβλεψη της. Επίσης ευχαριστώ ιδιαίτερα τον Νικόλαο Σπανό για την καθοδήγησή και υποστήριξή του σε όλη τη διάρκεια της συνεργασίας μας. Τέλος θα ήθελα να ευχαριστήσω τους γονείς μου για την κατανόηση που μου έδειξαν και τη συμπαράσταση που μου προσέφεραν όλα αυτά τα χρόνια.

Αθήνα, Ιούλιος 2025

Αλέξανδρος Συριγωνάκης

Περιεχόμενα

Κατάλογος Εικόνων

Κατάλογος Πινάκων

Μέρος Ι

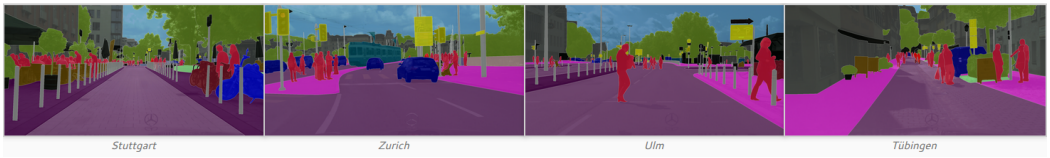
Εκτεταμένη Περίληψη στα Ελληνικά

Κεφάλαιο 1

Θεωρητικό υπόβαθρο

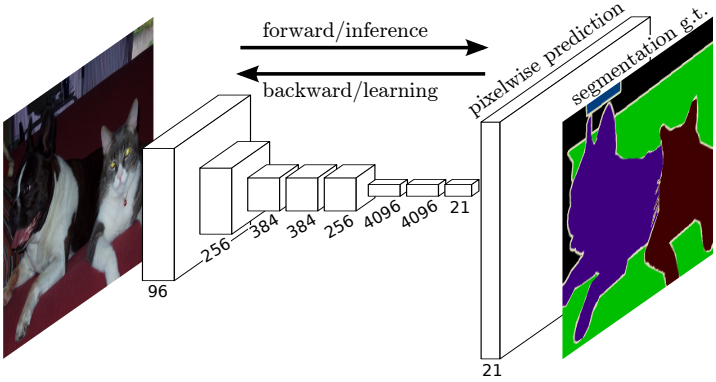
1.1 Κατάτμηση Εικόνων

Η κατάτμηση εικόνων αποτελεί μία θεμελιώδη εφαρμογή της Όρασης Υπολογιστών [15]. Μια εικόνα χωρίζεται σε πολλαπλά τμήματα, με κάθε ένα να αντιπροσωπεύει ένα διαφορετικό αντικείμενο. Σε συνδυασμό με την αναγνώριση αντικειμένων, η κατάτμηση εικόνων έχει εφαρμογές στην ανάλυση ιατρικών εικόνων [16], στα αυτόνομα οχήματα [17], παρακολούθηση κλπ. Παραδείγματα κατάτμησης παρουσιάζονται στην εικόνα 1.1.

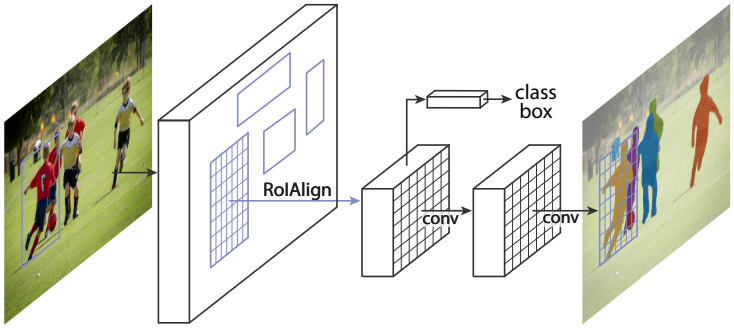


Εικόνα 1.1: *Παραδείγματα κατατμημένων εικόνων από αστικά περιβάλλοντα από [1]*

Γενικά ένα βαθύ νευρωνικό δίκτυο εκπαιδεύεται σε δεδομένα εικόνων και μαθαίνει να αποδίδει πυκνά μία ετικέτα σε κάθε εικονοστοιχείο. Αυτή η εργασία ταξινόμησης σε διαχωρισμένες περιοχές της αρχικής εικόνας με σημασιολογικές ετικέτες ονομάζεται σημασιολογική κατάτμηση (1.2). Μία επιπρόσθετη διαχώριση σε διακριτά αντικείμενα με την αναγνώριση πολλαπλών εμφανίσεων ενός αντικειμένου ονομάζεται κατάτμηση περιπτώσεων (1.3). Οι εργασίες κατάτμησης είναι σημαντικά πιο δύσκολες σε σύγκριση με απλές εργασίες ταξινόμησης, κάτι που αντανακλάται στις υπολογιστικές τους απαιτήσεις.



Εικόνα 1.2: *Παράδειγμα Σημασιολογικής Κατάτμησης με τη χρήση του μοντέλου FCN [2]*



Εικόνα 1.3: *Παράδειγμα Κατάτμησης Περιπτώσεων με τη χρήση του μοντέλου Mask R-CNN [3]*

1.2 Μηχανική Μάθηση

Η μηχανική μάθηση αφορά την ανάπτυξη αλγορίθμων που μαθαίνουν να εκτελούν κάποιες εργασίες και να γενικεύουν σε νέα δεδομένα χωρίς να τους έχουν δοθεί οδηγίες. Οι αλγόριθμοι στοχεύουν στην εκμάθηση μοτίβων ενσωματωμένων στα δεδομένα εκπαίδευσης για να προβλέπουν ή να αποφασίζουν βάσει της συσσωρευμένης τους γνώσης.

Οι αλγόριθμοι μηχανικής μάθησης παραδοσιακά χωρίζονται σε τρεις κατηγορίες, με βάση την φύση των εισόδων τους:

- **Επιβλεπόμενη Μάθηση**
- **Μη-Επιβλεπόμενη Μάθηση**
- **Ενισχυτική Μάθηση**

1.2.1 Επιβλεπόμενη Μάθηση

Στην επιβλεπόμενη μάθηση, ο αλγόριθμος εκπαιδεύεται σε ένα σύνολο δεδομένων που συμπεριλαμβάνει τη θεμιτή έξοδο. Τα δεδομένα εισόδου αποτελούνται από παραδείγματα εκπαίδευσης που αντιπροσωπεύονται από διανύσματα χαρακτηριστικών x. Κάθε παράδειγμα εκπαίδευσης έχει ένα ή περισσότερα δεδομένα επιθυμητής εξόδου y, που ονομάζονται επιβλεπόμενα σήματα. Η εκπαίδευση στοχεύει στην εύρεση μιας αντιστοίχισης $y = f(x)$ από την είσοδο στην έξοδο που μπορεί να κάνει ακριβείς προβλέψεις ή αποφάσεις σε νέα δεδομένα.

1.2.2 Μη Επιβλεπόμενη Μάθηση

Στην μη επιβλεπόμενη μάθηση τα δεδομένα εισόδου δεν περιέχουν κάποια πληροφορία για την επιθυμητή έξοδο. Ο σκοπός της εκπαίδευσης είναι η αναγνώριση δομών ή σχέσεων μεταξύ των δεδομένων.

Η αυτό-επιβλεπόμενη μάθηση είναι ένα είδος μη επιβλεπόμενης μάθησης, στην οποία το επιβλεπόμενο σήμα δημιουργείται από τον αλγόριθμο. Η ημι-επιβλεπόμενη μάθηση συνδυάζει επιβλεπόμενη με μη επιβλεπόμενη μάθηση, με το μοντέλο να εκπαιδεύεται τόσο σε δεδομένα με σήμανση όσο και σε δεδομένα χωρίς σήμανση.

1.2.3 Ενισχυτική Μάθηση

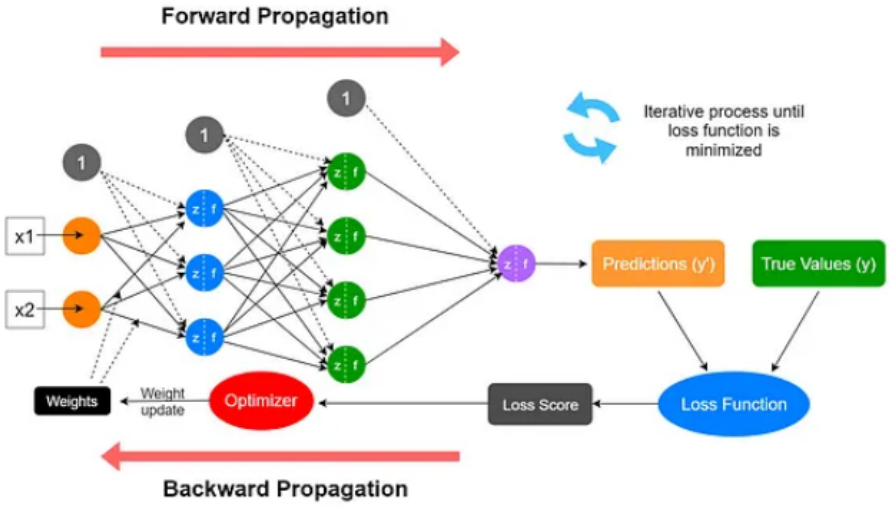
Στην ενισχυτική μάθηση ένα πράκτορας εκπαιδεύεται να παίρνει ακολουθιακές αποφάσεις, με στόχο την μεγιστοποίηση μίας μετρικής. Ένα τεχνητό περιβάλλον δημιουργείται για την εκπαίδευση του πράκτορα, ο οποίος αντιδρά στις συνθήκες και τις αλλαγές του περιβάλλοντος.

1.3 Νευρωνικά Δίκτυα

1.3.1 Τεχνητά Νευρωνικά Δίκτυα

Τα τεχνητά νευρωνικά δίκτυα (ΤΝΔ) είναι υπολογιστικά συστήματα που σχεδιάστηκαν βάσει βιολογικών νευρωνικών συστημάτων [6]. Ένα πλήθος διασυνδεδεμένων κόμβων, που ονομάζονται νευρώνες, σχηματίζουν αλλεπάλληλα επίπεδα. Κάθε επίπεδο επιτελεί μία συγκεκριμένη διεργασία, σύμφωνα με την αρχιτεκτονική του μοντέλου. Η βασική δομή ενός ΤΝΔ παρουσιάζεται στην εικονα 1.4, και αποτελείται από ένα επίπεδο εισόδου, έναν αριθμό κρυφών επιπέδων και ένα επίπεδο εξόδου που παρέχει το επιθυμητό αποτέλεσμα. Κάθε νευρώνας δέχεται σήματα εισόδου, τα οποία σταθμίζει, προσθέτει και λαμβάνει μία απόφαση, περνώντας το άθροισμα από μία συνάρτηση ενεργοποίησης. Το αποτέλεσμα οδηγείται στους συνδεδεμένους νευρώνες.

Η διαδικασία εκμάθησης περιλαμβάνει την αλλαγή των παραμέτρων βαρών των νευρώνων, με τον αλγόριθμο της οπισθοδιάδοσης, με στόχο την ελαχιστοποίηση κάποιας μετρικής διαφοράς μεταξύ επιθυμητού και πραγματικού αποτελέσματος, που ονομάζεται συνάρτηση απώλειας. Η επιλογή της συνάρτησης ενεργοποίησης και της συνάρτησης απώλειας εξαρτάται από την φύση της εργασίας.



Εικόνα 1.4: *Παράδειγμα Τεχνητού Νευρωνικού Δικτύου και παρουσίαση της διαδικασίας εκπαίδευσης [4]*

Η πιο συνηθισμένη συνάρτηση ενεργοποίησης είναι η ανορθωμένη γραμμική μονάδα (ReLU)

η οποία περιγράφεται ως:

$$f(x) = \max(0, x) \tag{1.1}$$

Η συνάρτηση είναι εύκολη στην υλοποίηση και αντιμετωπίζει με επιτυχία το πρόβλημα της εξαφανιζόμενης κλίσης.

Μία εναλλακτική της ReLU, η ReLU με διαρροή επιτρέπει μία μικρή κλίση όταν η μονάδα δεν είναι ενεργή, και έχει καλύτερη συμπεριφορά σε σύγκριση με την ReLU σε κάποια προβλήματα.

Η σιγμοειδής συνάρτηση λαμβάνει τιμές μεταξύ 0 και 1, και χρησιμοποιείται όταν το μοντέλο πρέπει να κάνει μία πρόβλεψη στην έξοδο.

$$\sigma(x) = \frac{1}{1 + e^{-x}} \tag{1.2}$$

Η συνάρτηση softmax μετατρέπει ένα εύρος εισόδου σε μια κατανομή πιθανοτήτων, με άθροισμα 1.

$$\sigma(z_i) = \frac{e^{z_i}}{\sum_{j=1}^{K} e^{z_j}}. \tag{1.3}$$

Χρησιμοποιείται σε περιπτώσεις ταξινόμησης, για να παράξει τις τιμές πρόβλεψης για κάθε κλάση.

Μία από τις πιο χρησιμοποιούμενες συναρτήσεις απώλειας είναι το Μέσο Τετραγωνικό Σφάλμα:

$$L = \frac{1}{N} \sum_{i=1}^{N} (y_i - f(x_i))^2 \tag{1.4}$$

όπου f είναι η συνάρτηση που θέλουμε νε προσεγγίσουμε και y_i οι τιμές του στόχου. Η μετρική αυτή χρησιμοποιείται σε εργασίες παλινδρόμισης, όταν η έξοδος του μοντέλου είναι μία αριθμητική τιμή.

Το Μέσο Απόλυτο Σφάλμα προτιμάται όταν υπάρχουν πολλές ακραίες τιμές στο σύνολο εκπαίδευσης:

$$L = \frac{1}{N} \sum_{i=1}^{N} |y_i - f(x_i)| \tag{1.5}$$

Το σφάλμα διασταυρούμενης εντροπίας χρησιμοποιείται σε εργασίες ταξινόμησης. Μετράει τη διαφορά μεταξύ της προβλεπόμενης και της πραγματικής κατανομής ετικετών κλάσης. Για M αριθμό κλάσεων το σφάλμα υπολογίζεται:

$$L = -\sum_{i=1}^{M} p_i \log(q_i) \tag{1.6}$$

Η Κάθοδος Κλίσης (Gradient Descent) είναι ο αλγόριθμος βελτιστοποίησης που χρησιμοποιείται πιο συχνά στα νευρωνικά δίκτυα για την ελαχιστοποίηση της συνάρτησης απώλειας. Η κάθοδος κλίσης αλλάζει τις παραμέτρους του μοντέλου, χρησιμοποιώντας την κλίση απώλειας.

$$\theta_{new} = \theta - \eta \nabla_\theta L(\theta) \tag{1.7}$$

με η να είναι ο ρυθμός εκπαίδευσης.

Μία παραλλαγή της καθόδου κλίσης, η Στοχαστική Κάθοδος Κλίσης λαμβάνει μόνο ένα δείγμα από τα παραδείγματα εκπαίδευσης για τον υπολογισμό της κλίσης απώλειας. Παρότι σε κάποιες περιπτώσεις δείχνει αστάθεια, η Στοχαστική Κάθοδος Κλίσης είναι σημαντικά πιο γρήγορη σε σχέση με την Κάθοδο Κλίσης.

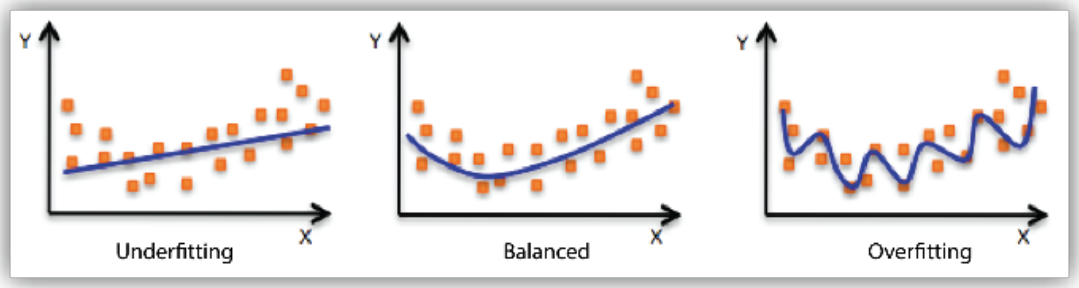
Ο αλγόριθμος Οπισθοδιάδοσης αποτελείται από δύο βήματα: έναν προς τα εμπρός υπολογισμό των προβλέψεων του μοντέλου και μία προς τα πίσω αλλαγή των παραμέτρων του μοντέλου, με τον υπολογισμό της κλίσης απώλειας για κάθε επίπεδο του μοντέλου.

Η διαδικασία εκπαίδευσης, με την πρόσθια πρόβλεψη και την οπίσθια αλλαγή των βαρών των νευρώνων, συνεχίζεται για πολλαπλές επαναλήψεις ή εποχές, βελτιώνοντας την ικανότητα του μοντέλου, μέχρι η διαδικασία να διακοπεί από κάποιο κριτήριο (για παράδειγμα μέγιστος αριθμός εποχών ή πολύ μικρό σφάλμα).

Η παράμετρος του ρυθμού εκπαίδευσης στην κάθοδο κλίσης είναι πολύ σημαντική για την εκπαιδευτική διαδικασία. Μία μικρή τιμή μπορεί να καθυστερήσει την σύγκλιση του μοντέλου, ή και να μην συγκλίνει ποτέ αν η εκπαίδευση έχει περιορισμένο αριθμό εποχών. Από την άλλη πλευρά, μία μεγάλη τιμή ρυθμού εκπαίδευσης μπορεί να οδηγήσει το μοντέλο στο να ξεπεράσει το ελάχιστο, ή και να ταλαντεύεται μεταξύ μη βέλτιστων θέσεων. Η επιλογή του ρυθμού εκπαίδευσης εξαρτάται από τη φύση κάθε προβλήματος.

Ένα μοντέλο βαθιάς μάθησης πρέπει να έχει την απαιτούμενη πολυπλοκότητα για να μπορεί να μάθει τα μοτίβα που εμπεριέχονται στα δεδομένα εκπαίδευσης. Η Υποπροσαρμογή συμβαίνει όταν η πολυπλοκότητα των δεδομένων προς εκπαίδευση υπερβαίνει την αντιληπτική ικανότητα του μοντέλου. Η Υπερπροσαρμογή συμβαίνει όταν το μοντέλο απομνημονεύει την πληροφορία του συνόλου εκπαίδευσης, χωρίς να διακρίνει μεταξύ χρήσιμων χαρακτηριστικών και θορύβου. Όταν ένα υπερπροσαρμοσμένο μοντέλο εφαρμόζεται σε νέα δεδομένα αποδίδει πολύ χειρότερα σε σχέση με την απόδοση του κατά την εκπαιδευτική διαδικασία.

Ένα απλό παράδειγμα προβλημάτων προσαρμογής παρουσιάζεται στο 1.5.



Εικόνα 1.5: *Παράδειγμα Υποπροσαρμογής και Υπερπροσαρμογής [5]*

Η υποπροσαρμογή προκαλείται από ένα μοντέλο που δεν έχει αρκετό βάθος, και μπορεί εύκολα να παρατηρηθεί όταν το σφάλμα είναι μεγάλο τόσο στα δεδομένα εκπαίδευσης, όσο και στα δεδομένα ελέγχου. Η υπερπροσαρμογή, από την άλλη, δεν μπορεί πάντα να γίνει αντιληπτή από την επίδοση του μοντέλου. Συνήθως η ευρωστία του μοντέλου στην υπερπροσαρμογή έχει σχέση και με την ικανότητα γενίκευσης του.

Οι μέθοδοι Κανονικοποίησης στοχεύουν στο να μειώσουν την επίδραση της υπερπροσα-

ρογής. Κατά τη διάρκεια της εκπαίδευσης, μία ποινή προστίθεται στη συνάρτηση απώλειας, με στόχο την μείωση της υπερβολικής πολυπλοκότητας. Οι δύο πιο συχνοί τρόποι κανονικοποίησης είναι η κανονικοποίηση L1 και L2. Στην L1 (Lasso) κανονικοποίηση ο όρος $\lambda_1 \sum_i^N |w_i|$ προστίθεται στην συνάρτηση απώλειας. Ενθαρρύνει αραιούς πίνακες βαρών, με τα περισσότερα βάρη να γίνονται 0. Οι νευρώνες τείνουν να χρησιμοποιούν μόνο ένα μέρος των εισόδων τους, που αντιστοιχουν στις πιο σημαντικές εισόδους για την εργασία, με αποτέλεσμα να υπάρχει ανθεκτικότητα στο θόρυβο. Στην L2 (Ridge) κανονικοποίηση ο όρος $\lambda_2 \sum_i^N |w_i{}^2|$ προστίθεται στην συνάρτηση απώλειας. Ενθαρρύνει μικρότερες τιμές βαρών και την αξιοποίηση όλων των εισόδων.

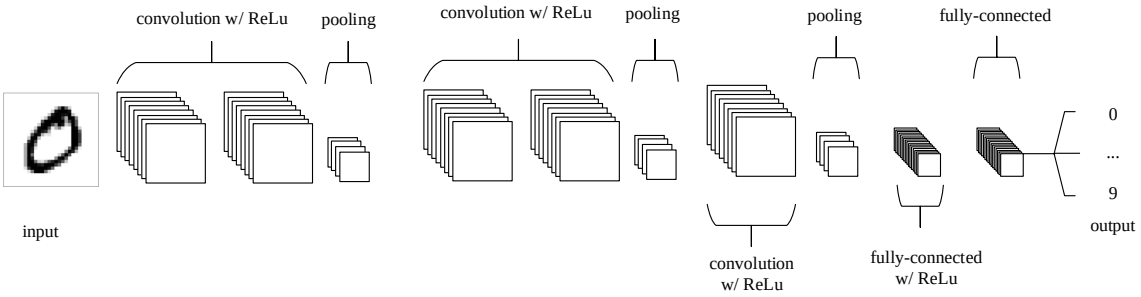
Οι μέθοδοι κανονικοποίησης μειώνουν την υπερπροσαρμογή, βελτιώνοντας την επίδοση και τη σταθερότητα των μοντέλων.

1.3.2 Συνελικτικά Νευρωνικά Δίκτυα

Τα κλασσικά ΤΝΔ έχουν περιορισμένη χρήση σε εργασίες Όρασης Υπολογιστών, καθώς δεν έχουν την απαιτούμενη πολυπλοκότητα για την επεξεργασία εικόνων. Επίσης επηρεάζονται από φαινόμενα υπερπροσαρμογής με την αύξηση των κρυφών επιπέδων που συνήθως απαιτείται σε εργασίες Όρασης Υπολογιστών.

Τα Συνελικτικά Νευρωνικά Δίκτυα (ΣΝΔ) (1.6) είναι ένα είδος νευρωνικών δικτύων εμπρόσθιας τροφοδότησης που εξάγουν χαρακτηριστικά από δεδομένα χρησιμοποιώντας συνελικτικές δομες [18]. Έχουν σχεδιαστεί κυρίως για χρήση σε εργασίες Όρασης Υπολογιστών. Τα κύρια πλεονεκτήματά τους, σε σχέση με τα ΤΝΔ είναι:

- Τοπικές συνδέσεις
- Διαμοιρασμός βαρών
- Μείωση διαστάσεων μέσω υποδειγματοληψίας

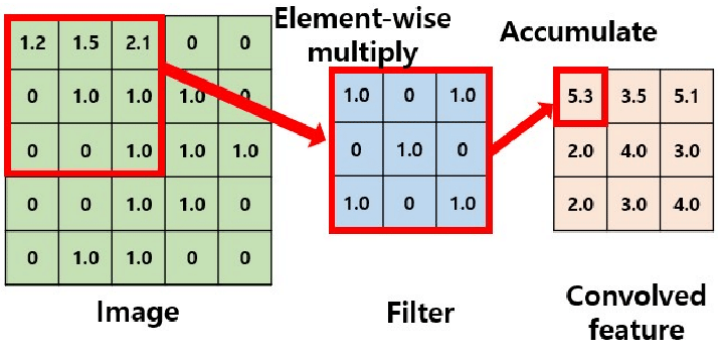


Εικόνα 1.6: *Παρουσίαση μιας απλής αρχιτεκτονικής Συνελικτικού Νευρωνικού Δικτύου [6]*

Ένα τυπικό ΣΝΔ έχει 3 βασικά είδη επιπέδων [6]: το επίπεδο συνέλιξης, το επίπεδο υποδειγματοληψίας και το πλήρως συνδεδεμένο επίπεδο.

Το επίπεδο Συνέλιξης υπολογίζει το γινόμενο μεταξύ των επιπέδων βαρών και της εισόδου, όπως φαίνεται στο 1.7. Οι παράμετροι των επιπέδων αντιστοιχούν σε συνελικτικούς πυρήνες, εκπαιδευόμενες δομές φίλτρων με μικρότερες χωρικές διαστάσεις, που εφαρμόζονται σε όλη την είσοδο για να δημιουργήσουν ένα χάρτη ενεργοποίησης, επισημάνοντας συγκεκριμένα

χαρακτηριστικά της εισόδου. Συγκρίνοντας με τα ΤΝΔ, οι νευρώνες των Συνελικτικών Δικτύων συνδέονται με μια μικρή περιοχή της εισόδου, που αναφέρεται ως το αντιληπτικό πεδίο του νευρώνα.



Εικόνα 1.7: *Παράδειγμα Συνέλιξης [7]*

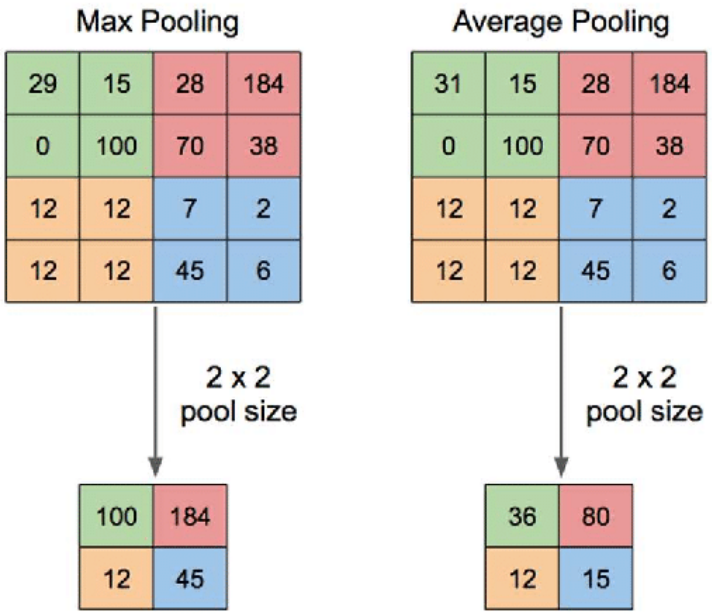
Τα συνελικτικά επίπεδα βελτιστοποιούνται με τις υπερπαραμέτρους βάθος, βήμα και γέμισμα. Το βάθος αναφέρεται στον αριθμό των νευρώνων που συνδέονται με την ίδια περιοχή της εισόδου. Η μείωση του βάθους οδηγεί σε σημαντική μείωση των παραμέτρων ενός μοντέλου, αλλά μπορεί να επιφέρει μείωση της επίδοσης. Το βήμα αναφέρεται στον αριθμό εικονοστοιχείων για τον οποίο μετακινείται το συνελικτικό φίλτρο μετά από κάθε πράξη, και επηρεάζει τις διαστάσεις του τελικού χάρτη ενεργοποίησης. Το γέμισμα εφαρμόζεται στις άκρες της εισόδου, συνήθως με μηδενικές τιμές, με στόχο τον έλεγχο των διαστάσεων της εξόδου.

Για κάθε διάσταση της εισόδου, η αντίστοιχη διάσταση της εισόδου μπορεί να βρεθεί από τον τύπο:

$$W_{out} = \frac{W_{in} - K + 2P}{S} + 1 \tag{1.8}$$

όπου K είναι η αντίστοιχη διάσταση του πυρήνα της συνέλιξης, S είναι η αντίστοιχη διάσταση του βήματος και P είναι η τιμή γεμίσματος.

Το επίπεδο υποδειγματοληψίας εφαρμόζεται στις χωρικές διαστάσεις της εισόδου, χρησιμοποιώντας μία δειγματοληπτική συνάρτηση όπως η μέγιστη ή η μέση τιμή. Ένα παράδειγμα παρουσιάζεται στο 1.8. Στα περισσότερα ΣΝΔ χρησιμοποιείται η δειγματοληπτική συνάρτηση του μέγιστου, με πυρήνες 2×2 να εφαρμόζονται στην είσοδο με βήμα 2, κλιμακώνοντας την αρχική εικόνα στο ένα τέταρτό της. Ως δειγματοληπτικές συναρτήσεις μπορούν να χρησιμοποιηθούν και μέθοδοι κανονικοποίησης.



Εικόνα 1.8: *Παράδειγμα υποδειγματοληπτικών λειτουργιών [8]*

Τα Πλήρως Συνδεδεμένα Επίπεδα περιέχουν νευρώνες που είναι πυκνά συνδεδεμένοι με τους νευρώνες των γειτονικών επιπέδων, παρομοιάζοντας τις συνδέσεις ενός ΤΝΔ. Συνήθως χρησιμοποιούνται ως επίπεδα εξόδου σε εργασίες ταξινόμησης.

Εκτός από τα κύρια επίπεδα, μπορούν να προστεθούν και άλλα λειτουργικά επίπεδά στην αρχιτεκτονική του ΣΝΔ.

Το επίπεδο κανονικοποίησης παρτιδας [19] είναι μια μέθοδος κανονικοποίησης που μπορεί να επιταχύνει την διαδικασία εκμάθησης. Χρησιμοποιούνται για να κανονικοποιήσουν τους χάρτες ενεργοποίησης κάθε παρτίδας. Για μία παρτίδα x, το επίπεδο θα δώσει έξοδο:

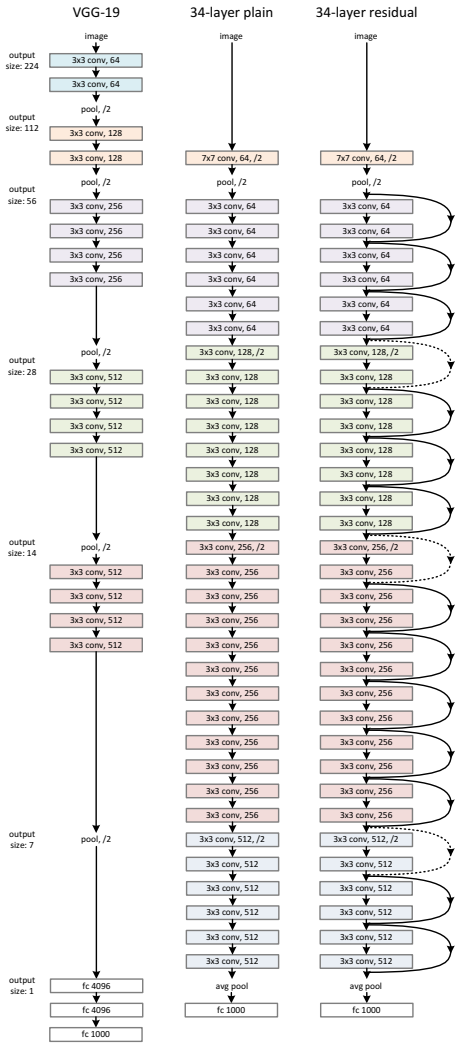
$$\hat{x}_i = \frac{x_i - \mu_B}{\sqrt{\sigma_B^2 + \epsilon}} \tag{1.9}$$

με τον μέσο και την διακύμανση να υπολογίζονται:

$$\mu_B = \frac{1}{m}\sum_{i=1}^{m} x_i \qquad \sigma_B^2 = \frac{1}{m}\sum_{i=1}^{m}(x_i - \mu_B)^2 \tag{1.10}$$

Η ϵ αρχικοποιείται με μία μικρή τιμή για να αποφευχθεί η διαίρεση με 0.

Η τεχνική απόρριψης [20] διαλέγει τυχαία ένα συγκεκριμένο ποσοστό από τους νευρώνες, με τις συνδέσεις τους να απορρίπτονται από το δίκτυο. Τα επίπεδα απόρριψης μειώνουν σημαντικά την υπερπροσαρμογή και είναι πιο αποδοτικά σε σχέση με άλλες μεθόδους κανονικοποίησης.



Εικόνα 1.9: *Αρχιτεκτονική του μοντέλου ResNet34*

1.4 Υπολλειμματικά Συνελικτικά Δίκτυα

Το βάθος των συνελικτικών νευρωνικών δικτύων είναι σημαντικό για την επίδοση των μοντέλων σε εργασίες ταξινόμησης εικόνων. Ωστόσο, το βάθος μπορεί να οδηγήσει σε φαινόμενα εξαφάνισης κλίσης ή έκρηξης κλίσης, που επηρεάζουν αρνητικά την εκπαιδευτική διαδικασία, με την επίδοση του μοντέλου να εκδηλώνει κορεσμό ή και μείωση. Αυτό το φαινόμενο υποβάθμισης υποδηλώνει ότι η προσέγγιση ταυτοτικών χαρτογραφήσεων από πολλαπλά μη γραμμικά επίπεδα μπορεί να είναι δύσκολη για τα δίκτυα. Συνήθως τα προβλήματα αυτά αντιμετωπίζονταν με επαρκώς μικρούς ρυθμούς μάθησης, κανονικοποιημένη αρχικοποίηση των βαρών ή και ενδιάμεσα επίπεδα κανονικοποίησης. Για την βελτίωση της επίδοσης των βαθιών Συνελικτικών Δικτύων, οι συγγραφείς του [21] προτείνουν την αναδιαμόρφωση των επιπέδων συνέλιξης ως εκπαιδευόμενες υπολλειμματικές συναρτήσεις αναφερόμενες στο επίπεδο εισόδου. Με την υπολλειμματική μάθηση, αν η ταυτοτική χαρτογράφιση ενός επιπέδου είναι βέλτιστη, τότε τα βάρη του θα παραβλέπονται, αποκτώντας μηδενική τιμή.

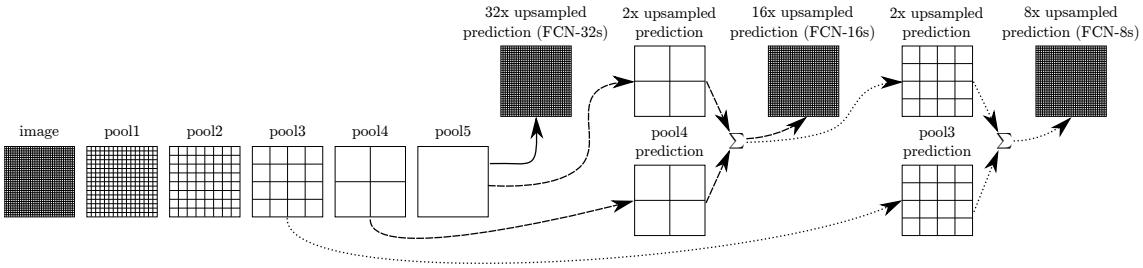
Έχοντας έναν αριθμό στοιβαγμένων επιπέδων και την επιθυμητή χαρτογράφηση $\mathcal{H}(x)$ προς προσαρμογή, το στοιβαγμένο μη γραμμικό επίπεδο θα προσαρμοστεί στη χαρτογράφη-

ση $\mathcal{F}(x) := \mathcal{H}(x) - x$, θεωρώντας ότι είσοδος και έξοδος έχουν τις ίδιες διαστάσεις. Η αρχική συνάρτηση γίνεται $\mathcal{F}(x) + x$. Η υλοποίηση μπορεί να γίνει χρησιμοποιώντας εμπρόσθιες συνδέσεις παράκαμψης, περνώντας έναν αριθμό από επίπεδα, χωρίς να προστίθενται παράμετροι στο δίκτυο. Η αρχιτεκτονική ResNet-34 παρουσιάζεται στην εικόνα 1.9.

Τα υπολλειμματικά δίκτυα βελτιστοποιούνται πιο εύκολα, συγκρινώμενα με δίκτυα παρόμοιου βάθους.

1.5 Πλήρως Συνελικτικά Δίκτυα

Τα πλήρως συνελικτικά δίκτυα [2] επεκτείνουν τις ικανότητες των ΣΝΔ στη σημασιολογική κατάτμηση. Στις εργασίες σημασιολογικής κατάτμησης, τόσο τοπικές, όσο και συνολικές πληροφορίες παίζουν ρόλο στη επίδοση του μοντέλου. Με την εξαγωγή χαρτογραφήσεων χαρακτηριστικών από πολλαπλά επίπεδα, τα πλήρως συνελικτικά δίκτυα μπορούν να συνδυάζουν βαθιές, αδρές σημασιολογικές πληροφορίες με ρηχές, λεπτές πληροφορίες εμφάνισης.

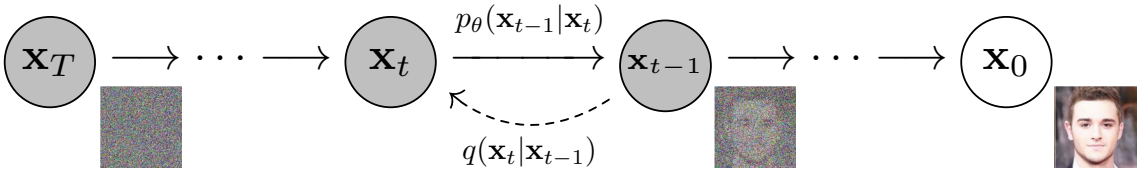


Εικόνα 1.10: *Οι τρεις βασικές αρχιτεκτονικές του Πλήρους Συνελικτικού Δικτύου*

Η δομή ενός Πλήρως Συνελικτικού Δικτύου (ΠΣΔ) παρουσιάζεται στο 1.10. Συνήθως ένα ΠΣΔ χρησιμοποιεί ένα βαθύ συνελικτικό δίκτυο ως σκελετό. Το δίκτυο του σκελετού ενσωματώνεται στο ΠΣΔ χωρίς τελικό επίπεδο ταξινόμησης. Στην έξοδο μια συνέλιξη 1×1 με διάσταση καναλιού ίση με τον επιθυμητό αριθμό της εργασίας ακολουθείται από ένα επίπεδο απο-συνέλιξης με στόχο την μετατροπή της αδρής εξόδου σε έξοδο εικονοστοιχείων. Η αρχιτεκτονική αυτή χρησιμοποιεί ένα βήμα 32 εικονοστοιχείων και ονομάστηκε FCN-32s. Για την βελτίωση ποιότητας της εικόνας εξόδου, επιπλέον συνδέσεις με προηγούμενα επίπεδα αξιοποιήθηκαν, με πιο λεπτά βήματα. Οι πιο περίπλοκες αρχιτεκτονικές είχαν καλύτερη επίδοση από το FCN-32s τόσο ποιοτικά όσο και ποσοτικά.

1.6 Μοντέλα Διάχυσης

Οι βασικές ιδέες, πάνω στις οποίες στηρίχθηκαν τα μοντέλα διάχυσης εισήχθηκαν από τους συγγραφείς του [22]. Τα μοντέλα αυτά ακολουθούν μία διαδικασία εμπρόσθιας διάχυσης κατά την οποία εισάγεται σταδιακά θόρυβος στην αρχική εικόνα, με στόχο την μετατροπη της αρχικής κατανομής σε μία κατανομή γνωστής συμπεριφοράς, και μία αντίστροφη διαδικασία γένεσης, στην οποία το μοντέλο μαθαίνει την διαδικασία ανασχηματισμού της αρχικής εικόνας. Η διαδικασία διάχυσης παρουσιάζεται στην εικόνα 1.11.



Εικόνα 1.11: *Παρουσίαση της διαδικασίας ανασχηματισμού της αρχικής εικόνας από ένα δίκτυο διάχυσης [9]*

Η εμπρόσθια διαδικασία χαρακτηρίζεται από την εξίσωση:

$$q(\mathbf{x}_{1:T}|\mathbf{x}_0) \coloneqq \prod_{t=1}^{T} q(\mathbf{x}_t|\mathbf{x}_{t-1}) \coloneqq \prod_{t=1}^{T} \mathcal{N}(\mathbf{x}_t;\ \sqrt{1-\beta_t}\mathbf{x}_{t-1}, \beta_t\mathbf{I}) \tag{1.11}$$

ενώ η αντίστροφη διαδικασία χαρακτηρίζεται από την εξίσωση:

$$p_\partial(\mathbf{x}_{0:T}) \coloneqq p(\mathbf{x}_T) \prod_{t=1}^{T} p_\partial(\mathbf{x}_{t-1}|\mathbf{x}_t) \coloneqq p(\mathbf{x}_T) \prod_{t=1}^{T} \mathcal{N}(\mathbf{x}_{t-1}; \mu_\partial(\mathbf{x}_t, t), \mathbf{\Sigma}_\partial(\mathbf{x}_t, t)) \tag{1.12}$$

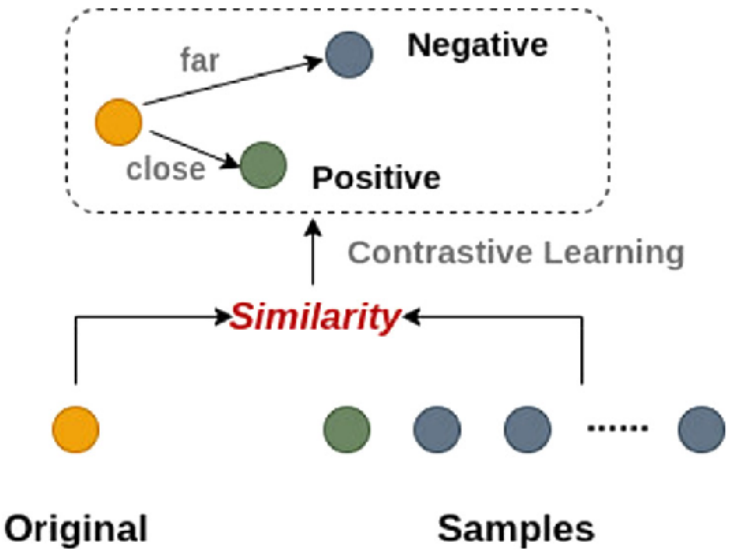
Η συνάρτηση απώλειας που χρησιμοποιείται για την εκπαίδευση του δικτύου είναι:

$$L_{VLB} \coloneqq D_{\mathrm{KL}}(q(\mathbf{x}_{t-1}|\mathbf{x}_t, \mathbf{x}_0) \parallel p_\partial(\mathbf{x}_{t-1}|\mathbf{x}_t)) \tag{1.13}$$

1.7 Αντικρουόμενη Μάθηση

Στόχος της Αντικρουόμενης Μάθησης [23, 24, 25] είναι η μάθηση όμοιων/ανόμοιων χαρακτηριστικών από δεδομένα ταξινομημένα σε αντίστοιχα ζεύγη. Βασίζεται στην υπόθεση ότι οι όμοιες παραστάσεις περιέχουν πληροφορίες που είναι κοντά μεταξύ τους, ενώ στις ανόμοιες παραστάσεις οι πληροφορίες απέχουν μεταξύ τους [26]. Η αντικρουόμενη μάθηση μπορεί πιο εύκολα να εντοπίσει σχετιζόμενα χαρακτηριστικά και ομοιότητες και να απομακρύνει ανόμοιες πληροφορίες. Ένα παράδειγμα Αντικρουόμενης μάθησης παρουσιάζεται στην εικόνα 1.12.

Συνήθως η αντικρουόμενη μάθηση συνδυάζεται με επαύξηση δεδομένων, αξιοποιώντας χωρικούς και χρωματικούς μετασχηματισμούς για την ενίσχυση του αρχικού συνόλου δεδομένων, βελτιώνοντας την αντοχή του μοντέλου σε διαφοροποιήσεις. Ενα δίκτυο κωδικοποιητή χρησιμοποιείται για τη δημιουργία της χαρτογράφησης του επαυξημένου συνόλου, και στη συνέχεια με τη χρήση ενός δικτύου προβολής μετατρέπεται σε μία απεικόνιση λιγότερων διαστάσεων. Η αντικρουόμενη διαδικασία εκτελείται σε αυτό το επίπεδο, που ονομάζεται επίπεδο ενσωματωμένων πληροφοριών, με τη χρήση μιας κατάλληλης συνάρτησης απώλειας.



Εικόνα 1.12: *Μια απεικόνιση του τρόπου λειτουργίας της Αντικρουόμενης Μάθησης [10]*

1.8 Προσαρμογή Πεδίου και Γενίκευση Πεδίου

Η μεταφορά στυλ στοχεύει στο να βοηθήσει ένα μοντέλο μηχανικής μάθησης να μεταφέρει την αποκτηθείσα γνώση από ένα αρχικό πεδίο σε ένα συγγενικό. Μία από τις πιο γνωστές μεθόδους μεταφοράς στυλ είναι η μέθοδος fine-tuning, κατά την οποία ένα προεκπαιδευμένο μοντέλο εκπαιδεύεται στοχευμένα για την προσαρμογή σε ένα εξειδικευμένο πεδίο. Η Προσαρμογή Πεδίου αποτελεί ένα είδος μεταφοράς στυλ, με την υπόθεση ότι μεταξύ των διαφορέτικών πεδίων αλλάζει μόνο η πληροφορία που σχετίζεται με το στυλ [27]. Στοχεύει στο να μετριάσει το πρόβλημα της αλλαγής πεδίου μεταξύ πηγής και προορισμού. [12].

Η Γενίκευση Πεδίου αποτελεί περίπτωση της προσαρμογής πεδίου, στην οποία δεν είναι γνωστό το πεδίο στο οποίο θέλουμε να προσαρμόσουμε το αρχικό. Στόχος της γενίκευσης πεδίου είναι η βελτίωση της ικανότητας γενίκευσης του συνόλου εκπαίδευσης σε οποιαδήποτε συνθήκη, και έχει πολλές εφαρμογές στην αναγνώριση αντικειμένων, στην σημασιολογική κατάτμηση και την αυτόνομη οδήγηση.

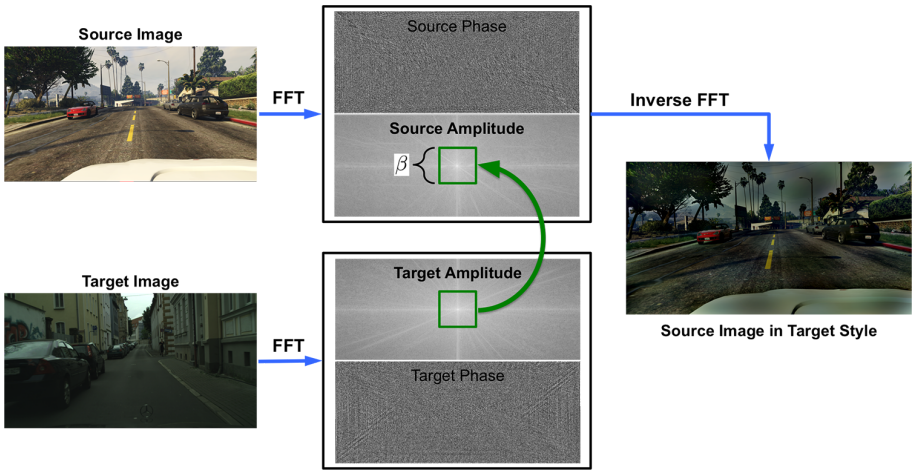
Κεφάλαιο 2

Προσέγγιση

2.1 Σχετική Βιβλιογραφία

2.1.1 Fourier Domain Adaptation (FDA)

Η μέθοδος FDA [14] είναι μία μέθοδος ημι-επιβλεπόμενης προσαρμογής πεδίου, που βασίζεται στην ελαχιστοποίηση της εντροπίας. Η διαπίστωση ότι το μεγαλύτερο μέρος της φωτομετρικής πληροφορίας μιας εικόνας βρίσκεται στο πεδίο χαμηλών συχνοτήτων αποτέλεσε το κίνητρο για αυτή τη μέθοδο. Αντικαθιστώντας αυτό το εύρος συχνοτήτων με τις συχνότητες του πεδίου που θέλουμε να προσεγγίσουμε μπορούμε να μεταφέρουμε ουσιώδη πληροφορία στο πεδίο εκπαίδευσης. Η μέθοδος χρησιμοποιεί τον αλγόριθμο FFT για να υπολογίσει τον μετασχηματισμό Φουριέ της εικόνας εισόδου, και στη συνέχεια αντικαθιστά τις συχνότητες χαμηλού πλάτους με τις αντίστοιχες συχνότητες από εικόνα του συνόλου προς προσέγγιση. Στο τέλος η αρχική εικόνα ανακατασκευάζεται με τον αντίστροφο μετασχηματισμό Φουριέ. Μία παράμετρος β χρησιμοποιείται ως όριο για την επιλογή του εύρους αντικατάστασης. Η μέθοδος παρουσιάζεται στην εικόνα 2.1.



Εικόνα 2.1: *Ο αλγόριθμος FDA*

Η εκπαίδευση γίνεται με τις ανακατασκευασμένες εικόνες και τις αρχικές τους επισημειώσεις.

2.1.2 The Saliency-balancing Location-scale Augmentation (SLAug)

Η μέθοδος Saliency−balancing Location−scale Augmentation (SLAug) [28] συνδυάζει τοπική και συνολική επαύξηση της εικόνας. Προηγούμενες μέθοδοι εφάρμοζαν μετασχηματισμούς είτε στο σύνολο της εικόνας είτε σε τμήματα της, χωρίς κάποια αλληλεπίδραση μεταξύ των τοπικών χαρακτηριστικών με την συνολική εικόνα, κάτι που είχε αρνητική επίδραση στην ικανότητα γενίκευσης των μοντέλων. Αυτή η πιο ολοκληρωμένη μέθοδος επαύξησης συνδυάζεται με έναν μηχανισμό εξισορρόπησης της κατανομής κλίσης.

Η επαύξηση στο επίπεδο περιοχής γίνεται σε δύο τμήματα. Η παγκόσμια επαύξηση στο επίπεδο περιοχής (GLA) επαυξάνει τις εικόνες που είναι πιο κοντά στο αρχικό πεδίο, χρησιμοποιώντας μια παγκόσμια μεταβολή της κατανομής τους. Η κυβική καμπύλη Bézier [29], μία ομαλή και μονότονη συνάρτηση, χρησιμοποιείται για τον μη γραμμικό μετασχηματισμό της εικόνας. Η συνάρτηση δεσμεύεται από την μέγιστη (v_{high}) και ελάχιστη (v_{low}) τιμή της περιοχής εικόνων.

$$Bézier(t) = \sum_{k=0}^{3} (1-t)^{3-k} t^{k P_k}, t \in [v_{\text{low}}, v_{\text{high}}], \tag{2.1}$$

όπου t είναι μια κλασματική τιμή, $P_0 = (v_{\text{low}}, v_{\text{low}})$ και $P_3 = (v_{\text{high}}, v_{\text{high}})$ είναι τα αρχικά και τελικά σημεία για τον περιορισμό του εύρους τιμών και P_1, P_2 είναι τα σημεία ελέγχου, οι τιμές των οποίων δημιουργούνται τυχαία στο $[v_{\text{low}}, v_{\text{high}}]$. Πριν την επαύξηση, η x κανονικοποιείται στο εύρος [0, 1]. Η GLA μπορεί να περιγραφεί:

$$\text{GLA}(x) = a\mathcal{F}_0(x) + \beta, \tag{2.2}$$

όπου $a \sim \mathcal{TN}(1, \sigma_1)$, $\beta \sim \mathcal{TN}(0, \sigma_2)$ είναι παράγοντες επιπέδου περιοχής και σ_1, σ_2 είναι οι τυπικές αποκλίσεις των δύο αποκομμένων κανονικών κατανομών. Χρησιμοποιώντας το $\mathcal{F}_0$, η πιθανότητα της αντιστρόφου είναι μηδέν ώστε να διασφαλιστεί ότι τα δείγματα GLA έχουν παρόμοια εμφάνιση με τις αρχικές εικόνες.

Η τοπική επαύξηση στο επίπεδο περιοχής (LLA) λαμβάνει τις ξεχωριστές περιοχές επιπέδου κλάσης ως τη μονάδα επεξεργασίας και εφαρμόζει τον μετασχηματισμό. Οι επαυξημένες περιοχές συνδυάζονται γραμμικά. Η διαδικασία αυτή μπορεί να αναπαρασταθεί:

$$\text{LLA}(x, m) = \sum_{c=1}^{C} a_c \mathcal{F}_{\text{p}_c}(x^c) + \beta_c, \tag{2.3}$$

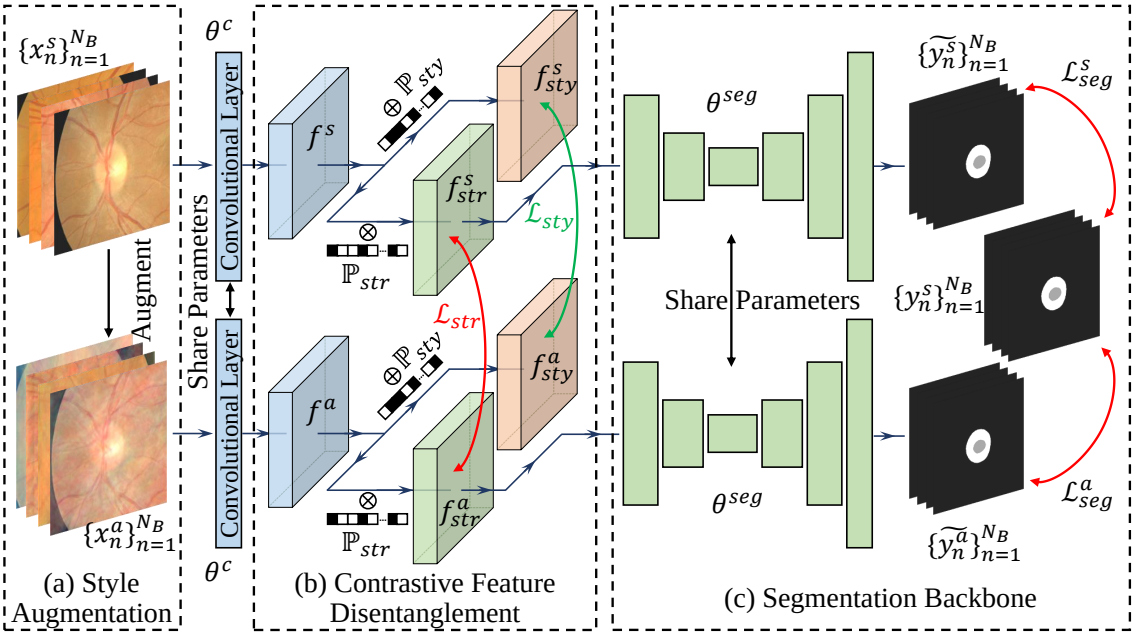
όπου $a_c \sim \mathcal{TN}(1, \sigma_1)$ και $\beta_c \sim \mathcal{TN}(0, \sigma_2)$ είναι παράγοντες επιπέδου περιοχής. Για όλες τις κλάσεις τίθεται $\text{p}_c = 0.5$ για να εφαρμοστεί τυχαία αντιστροφή, με εξαίρεση την $\text{p}_1 = 1$ για να διασφαλιστεί ότι οι επαυξημένες εικόνες του LLA θα είναι ανόμοιες με τα δείγματα του GLA.

Η σύμπτυξη με βάσει την εξισορρόπηση των χαρακτηριστικών στοχεύει στο να διατηρήσει τις περιοχές με μεγάλη κλίση, καθώς αυτές υποδεικνύουν πληροφοριακή ευαισθησία.

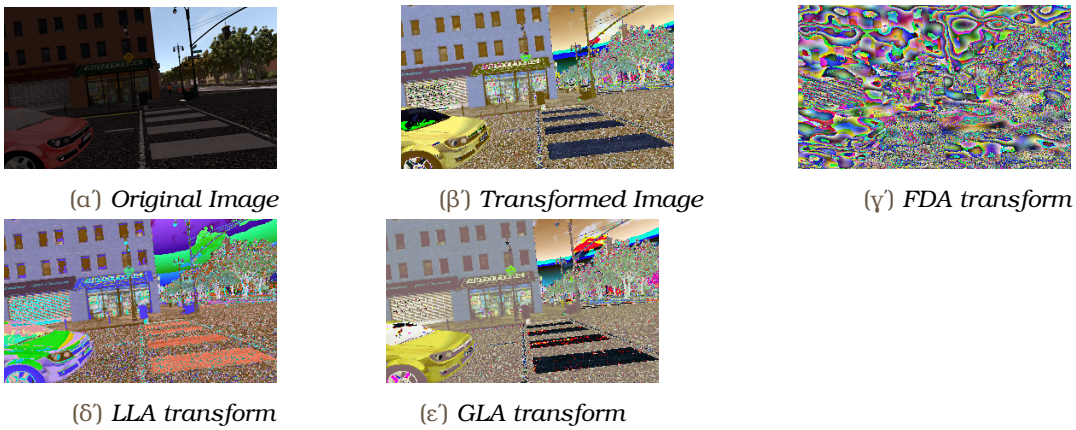
2.1.3 Channel-Level Contrastive Single Domain Generalization (CCSDG)

Το κύριο πρόβλημα της Γενίκευσης Ενός Πεδίου είναι δυσκολία γενίκευσης στο πεδίο-στόχο, που προκαλείται από την υπερπροσαρμογή των μοντέλων στο πηγαίο πεδίο. Η

μέθοδος CCSDG [30] υλοποιεί μια προσέγγιση αντικρουόμενης μάθησης για να επιτύχει καλύτερη επίδοση συγκρίνοντας με άλλες μεθόδους. Χρησιμοποιώντας τα πιο ρηχά χαρακτηριστικά των αρχικών εικόνων και τις αντίστοιχες εικόνες που έχουν δεχθεί επαύξηση το τμήμα απεμπλοκής χαρακτηριστικών καναλιών (Channel Feature Disentanglement) μαθαίνει να διακρίνει μεταξύ χαρακτηριστικών σχετιζόμενων με τη δομή της εικόνας, τα οποία δεν μεταβάλλονται στα άλλα πεδία και χαρακτηριστικών μη σχετιζόμενων με τη δομή της εικόνας, τα οποία είναι χαρακτηριστικά εγγενή σε κάθε πεδίο και δεν συνεισφέρουν στην ικανότητα γενίκευσης του μοντέλου.



Εικόνα 2.2: *Αρχιτεκτονική του CCSDG μοντέλου*



(α') *Original Image* (β') *Transformed Image* (γ') *FDA transform*

(δ') *LLA transform* (ε') *GLA transform*

Εικόνα 2.3: *Παράδειγμα εξόδου του τμήματος επαύξησης δεδομένων, χρησιμοποιώντας μία εικόνα από το σύνολο δεδομένων SYNTHIA [11]*

Η μέθοδος αυτή χρησιμοποιεί

- Ένα μοντέλο σκελετό για την κατάτμηση της εικόνας
- Ένα τμήμα επαύξησης στυλ (StyleAug) που δημιουργεί διαφοροποιήσεις στα στυλ των

εικόνων εισόδου

- Το τμήμα απεμπλοκής χαρακτηριστικών καναλιών που εκτελεί αντικρουόμενη μάθηση μεταξύ των χαρακτηριστικών των εικόνων εισόδου, τα οποία εξάγονται από ένα συνελικτικό επίπεδο, και των αντίστοιχων χαρακτηριστικών των επαυξημένων εικόνων

Η αρχιτεκτονική του CCSDG παρουσιάζεται στην εικόνα 2.2. Το τμήμα επαύξησης στυλ χρησιμοποιεί προσεγγίσεις όπως τη διόρθωση της παραμέτρου γάμμα και την εισαγωγή θορύβου από το [31], τους μετασχηματισμούς καμπύλης Bézier από το [28] και την αντικατάσταση τμημάτων χαμηλών συχνοτήτων από το [14] για να δημιουργήσει τρεις επαυξημένες παρτίδες από κάθε παρτίδα εισόδου για την αντικρουόμενη μάθηση.

Παράδειγμα της εξόδου του τμήματος επαύξησης στυλ παρουσιάζεται στις εικόνες 2.3

Το τμήμα απεμπλοκής χαρακτηριστικών καναλιών χρησιμοποιεί ένα συνελικτικό επίπεδο για να εξάγει από τις εικόνες εισόδου και τις επαυξημένες εικόνες, χάρτες χαρακτηριστικών 64 καναλιών. Με τη χρήση μιας προτρεπτικής μάσκας $\mathbb{P} \in \mathbb{R}^{2x64}$, η οποία έχει δομηθεί ώστε να ομοιάζει με διάνυσμα δυαδικών στοιχείων. κάθε χάρτης χαρακτηριστικών διαχωρίζεται σε μία αναπαράσταση στυλ f_{sty} και μία αναπαράσταση δομής f_{str}. Για την αντικρουόμενη εκπαίδευση, οι αναπαραστάσεις στυλ μεταξύ αρχικών και επαυξημένων εικόνων αναμένεται να είναι ανόμοιες ενώ οι αναπαραστάσεις δομής αναμένονται να είναι όμοιες. Δύο συναρτήσεις απώλειας έχουν σχεδιαστεί:

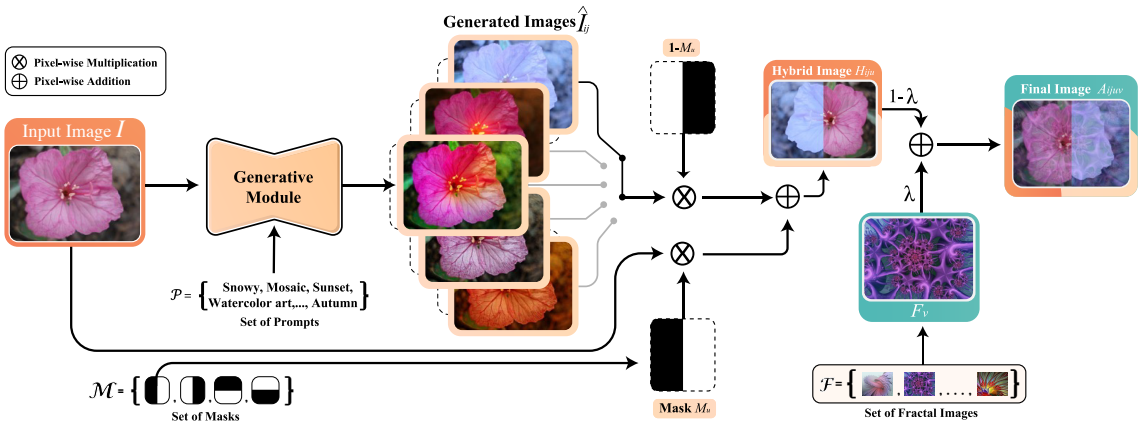
$$\begin{cases} \mathcal{L}_{str} = \sum |Proj(f^s_{str}) - Proj(f^a_{str})| \\ \mathcal{L}_{sty} = -\sum |Proj(f^s_{sty}) - Proj(f^a_{sty})|, \end{cases} \tag{2.4}$$

όπου το ^{s} αναφέρεται στους χάρτες χαρακτηριστικών της εισόδου και το ^{a} αναφέρεται στους χάρτες των επαυξημένων εικόνων. Το δίκτυο προβολής $Proj$, με παράμετρους ∂^p, μειώνει τις διαστάσεις των χαρτών χαρακτηριστικών για την αντιθετική μάθηση.

Οι αναπαραστάσεις δομής τροφοδοτούνται στο μοντέλο-σκελετό κατάτμησης ∂^{seg} για να παράξουν τις προβλέψεις. Με αυτό τον τρόπο, κατά την εκπαίδευση η $\mathcal{L}_{seg}$ ελαχιστοποιείται για την βελτιστοποίηση του $\{\partial^c, \mathbb{P}, \partial^{seg}\}$, και η $\mathcal{L}_{str} + \mathcal{L}_{sty}$ ελαχιστοποιείται για την βελτιστοποίηση του $\{\mathbb{P}, \partial^p\}$.

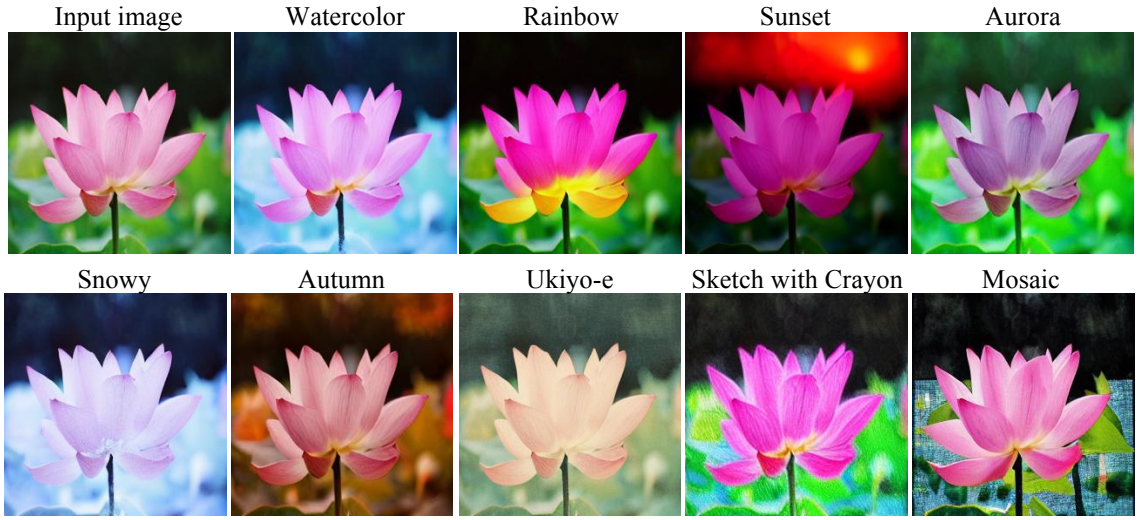
2.1.4 DiffuseMix

Η μέθοδος DiffuseMix στοχεύει στην επαύξηση δεδομένων μέσω ενός μοντέλου διάχυσης, για την ενίσχυση ενός αρχικού συνόλου δεδομένων. Η επαύξηση των εικόνων ακολουθεί τρία βήματα: δημιουργία, συνένωση και μίξη με επιλεγμένες εικόνες φράκταλ. Μία παρουσίαση του αλγορίθμου προυσιάζεται στο 2.4. Σε πρώτο στάδιο ένα προεκπαιδευμένο μοντέλο διάχυσης δημιουργεί ποικίλα δείγματα από τις αρχικές εικόνες, οδηγούμενο από μία επιλογή προτρεπτικών λέξεων. Στη συνέχεια οι εικόνες που δημιουργήθηκαν συνδυάζονται με τις αρχικές, χρησιμοποιώντας ένα σύνολο μασκών που συνενώνουν ένα τμήμα της μίας εικόνας με το τμήμα που απομένει από την άλλη, έτσι ώστε η τελική εικόνα να αποτελείται από δύο μισά των αρχικών. Στο τελευταίο βήμα η συνενωμένη εικόνα αναμιγνύεται με μία τυχαία εικόνα από ένα σύνολο αυτο-όμοιων φρακταλ.



Εικόνα 2.4: *Η αρχιτεκτονική DiffuseMix*

Αυτή η προσέγγιση διατηρεί τα βασικά χαρακτηριστικά της αρχικής εικόνας και παρέχει τη συμφραζόμενη πληροφορία που απαιτείται για πιο εύστοχη επάυξηση. Στο βήμα της δημιουργίας εικόνων χρησιμοποιήθηκε το μοντέλο InstructPix2Pix [32]. Το σύνολο των προτρεπτικών λέξεων επιλέχθηκε για την γενική του φύση και την μικρή παρεμβολή τους με τη βασική δομή των εικόνων (εικόνα 2.5).



Εικόνα 2.5: *Παράδειγμα της εξόδου του μοντέλου διάχυσης, με τις αντίστοιχες προτρεπτικές λέξεις*

Η μέθοδος DiffuseMix αποδείχθηκε ότι είναι ικανή να ενισχύσει την ικανότητα γενίκευσης του ResNet-50, παρουσιάζοντας συνεπή βελτίωση της επίδοσης σε πολλά σύνολα δεδομένων.

2.2 Μεθοδολογία

2.2.1 Αρχικό σύνολο δεδομένων

Για την εκπαίδευση χρησιμοποιήσαμε το σύνολο δεδομένων SYNTHIA, ένα κατάλογο από επισημασμένες εικόνες εικονικών αστικών περιβάλλοντων. Κάθε εικόνα είναι επισημασμένη σε επίπεδο εικονοστοιχείου σε 13 κλάσεις. Δημιουργήσαμε τα σύνολα εικόνων για

την εκπαίδευση και τον έλεγχο από τα σύνολα δεδομένων SYNTHIA-SEQS, τα οποία είναι ακολουθίες εικόνων από τέσσερις προομοιώσεις βίντεο, τα οποία λαμβάνονται από τη σκοπιά ενός εικονικού αυτοκινήτου. Επιλέξαμε τη OMNI-L σκοπιά για τα πειράματά μας. Από τα 5 διαθέσιμα περιβάλλοντα (Highway, New York City, Old European Town, New York City (2), Highway (2)) δημιουργήσαμε τρία σύνολα δεδομένων για την εκπαίδευση από το πρώτο περιβάλλον, παίρνοντας 800 τυχαίες εικόνες από τις συνθήκες (DAWN, FOG, SPRING), και 20 σύνολα δεδομένων για έλεγχο από τα υπόλοιπα 4 περιβάλλοντα, παίρνοντας 240 εικόνες από τις συνθήκες (DAWN, FOG, NIGHT, SPRING, WINTER). Οι εικόνες και οι ετικέτες τους μετασχηματίστηκαν από το αρχικό μέγεθος 1280×760 σε μέγεθος 320×192, με την συνάρτηση cv2.resize η οποία παρέχεται από την βιβλιοθήκη OpenCV της python, με παράμετρο παρεμβολής την τιμή cv2.INTER_NEAREST.

2.2.2 Σύνολο δεδομένων εκπαίδευσης

Χρησιμοποιήσαμε το προεκπαιδευμένο μοντέλο DiffuseMix [33] για να δημιουργήσουμε επαυξημένες παραλλαγές των αρχικών δεδομένων. Για κάθε μία από τις 3 συνθήκες εκπαίδευσης εφαρμόσαμε ένα υποσύνολο των παρεχόμενων από το μοντέλο προτρεπτικών λέξεων. Χρησιμοποιήσαμε τις generated εικόνες εξόδου του InstructPix2Pix [32] μοντέλου διάχυσης και τις blended εικόνες που είναι το αποτέλεσμα της μίξης των εικόνων που προέρχονται από τη συνένωση των αρχικών εικόνων με τις generated εικόνες με τυχαίες εικόνες φράκταλ. Χρησιμοποιώντας μία εικόνα από το αρχικό σύνολο δεδομένων, παρουσιάζουμε τις εξόδους στις εικόνες 2.6, 2.7, 2.8, 2.9

Εικόνα 2.6: *Παράδειγμα Αρχικής, Generated και Blended εξόδου εικόνας του DiffuseMix μοντέλου με χρήση της προτρεπτικής λέξης autumn*

Εικόνα 2.7: *Παράδειγμα Αρχικής, Generated και Blended εξόδου εικόνας του DiffuseMix μοντέλου με χρήση της προτρεπτικής λέξης snowy*

Εικόνα 2.8: *Παράδειγμα Αρχικής, Generated και Blended εξόδου εικόνας του DiffuseMix μοντέλου με χρήση της προτρεπτικής λέξης sunset*

Εικόνα 2.9: *Παράδειγμα Αρχικής, Generated και Blended εξόδου εικόνας του DiffuseMix μοντέλου με χρήση της προτρεπτικής λέξης ukiyo-e*

2.2.3 Μοντέλο Εκπαίδευσης

Το μοντέλο που χρησιμοποιήσαμε είναι μία τροποποιημένη έκδοση του προεκπαιδευμένου fcn_resnet50 που παρέχεται απο την βιβλιοθήκη torchvision.models. Το μοντέλο αποτελείται από ένα σκελετό resnet50 [21] με έναν ταξινομητή FCN−32s [2]. Προσαρμόσαμε το μοντέλο ώστε να δεχθεί το τμήμα απεμπλοκής χαρακτηριστικών καναλιών [30] μετά από το πρώτο συνελικτικό επίπεδο του μοντέλου-σκελετού.

Χρησιμοποιήσαμε επίσης το τμήμα Projector, ως μέρος της διαδικασίας αντικρουόμενης μάθησης που περιγράφεται στο [30].

2.2.4 Παράμετροι Εκπαίδευσης

Η διαδικασία εκπαίδευσης τίθεται να διαρκεί 120 εποχές, με μέγεθος παρτίδας ίσο με 8. Ο βελτιστοποιητής Adam επιλέχθηκε τόσο για το μοντέλο, όσο και για το τμήμα Projector, με τον ρυθμό εκμάθησης να τίθεται στην τιμή 10^{-4} και για τους δύο βελτιστοποιητές. Για τον βελτιστοποιητή του μοντέλου επιλέχθηκε φθορά βαρών ίση με 10^{-5}.

Το κριτήριο l1_loss [34] με την παράμετρο reduction να έχει την τιμή mean χρησιμοποιήθηκε για την εκπαίδευση του τμήματος απεμπλοκής χαρακτηριστικών καναλιών, όπως στο [30]. Αυτό το κριτήριο υπολογίζει το μέσο απόλυτο σφάλμα.

Για την εκπαίδευση του μοντέλου επιλέχθηκε το κριτήριο CrossEntropyLoss [35], με τις προτεινόμενες παραμέτρους. Η τιμή απώλειας για κάθε βήμα υπολογίζεται από τον τύπο:

$$l(x, y) = \sum_{n=1}^{N} \frac{l_n}{N} \qquad l_n = -\log \frac{\exp(x_{n,y_n})}{\sum_{c=1}^{C} \exp(x_{n,y_n})} \tag{2.5}$$

όπου Σ είναι ο αριθμός των κλάσεων και N το μέγεθος της παρτίδας.

Για την αξιολόγηση του μοντέλου χρησιμοποιήσαμε το κριτήριο Mean Intersection Over Union, το οποίο ορίζεται ως τη μέση τιμή διαίρεσης της τομής μεταξύ της πρόβλεψης και πραγματικής ετικέτας κλάσης και της ένωσης τους.

$$J(A, B) = \frac{|A \cap B|}{|A \cup B|} \tag{2.6}$$

Για τον υπολογισμό της μετρικής χρησιμοποιήσαμε την MulticlassJaccardIndex [36] συνάρτηση της βιβλιοθήκης torchmetrics, με τις προτεινόμενες παραμέτρους και αγνοώντας την ετικέτα παρασκηνίου.

2.2.5 Εκπαίδευση

Εκπαιδεύουμε το μοντέλο ακολουθώντας τα βήματα που περιγράφονται στο [30]. Στο τμήμα επαύξησης στυλ παραβλέπουμε την διόρθωση της παραμέτρου γάμμα και τους μετασχηματισμούς θορύβου.

Κάθε εποχή είχε διάρκεια περίπου 30 δευτερόλεπτα για το απλό μοντέλο και περίπου 120 δευτερόλεπτα για το επαυξημένο μοντέλο. Τα πειράματα εκτελέστηκαν σε περιβάλλον cloud computing που παρέχεται από τις Amazon Web Services, χρησιμοποιώντας ως επιταχυντή την NVIDIA A10G.

Κεφάλαιο 3

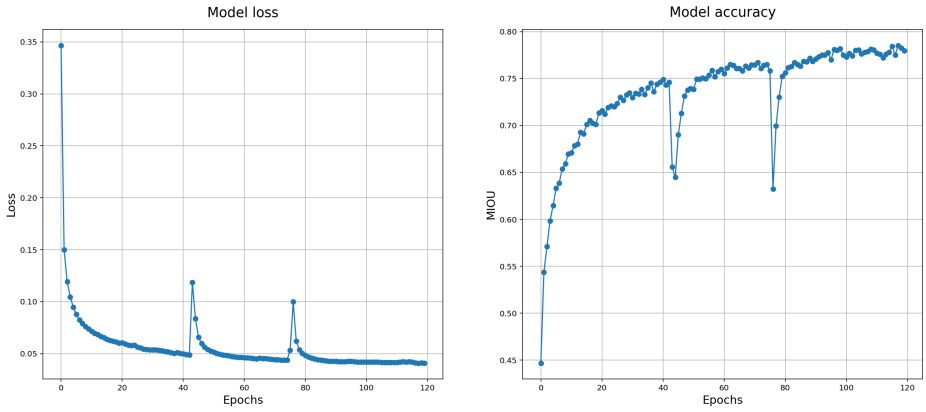
Πειράματα και Συμπεράσματα

3.1 Αποτελέσματα από την εκπαίδευση του βασικού μοντέλου

Πρώτα εκπαιδεύουμε το μοντέλο fcn_resnet50 χωρίς κάποια από τις αλλαγές που αναφέρθηκαν, στα σύνολα δεδομένων που δημιουργήθηκαν από το μοντέλο DiffuseMix

Σημειώνουμε ως baseline την επίδοση του μοντέλου που εκπαιδεύτηκε στο αρχικό σύνολο δεδομένων. Στην στήλη 'mode' σημειώνουμε ως 'gen' την Generated έξοδο και ως 'bl' την Blended έξοδο του DiffuseMix, με τις αντίστοιχες προτρεπτικές λέξεις. Οι υπόλοιπες στήλες αντιστοιχούν στα αποτελέσματα ελέγχου για κάθε συνθήκη του πεδίου.

Μία παρουσίαση της τιμής απώλειας και επίδοσης της εκπαίδευσης παρουσιάζεται στην εικόνα 3.1. Η επίδοση κάθε επιλογής παρουσιάζεται στους πίνακες 3.1, 3.2, 3.3, 3.4.



Εικόνα 3.1: *Διάγραμμα της απώλειας και της μετρικής MIOU στην εκπαίδευση*

New York City						
Train Domain	Mode	DAWN	FOG	NIGHT	SPRING	WINTER
DAWN	baseline	**0.305**	**0.282**	**0.249**	**0.288**	**0.268**
	gen–autumn	0.264	0.241	0.224	0.241	0.231
	gen–sunset	0.255	0.246	0.202	0.241	0.221
	gen–ukiyo-e	0.256	0.232	0.209	0.245	0.228
	bl–autumn	0.243	0.215	0.184	0.218	0.193
	bl–sunset	0.239	0.212	0.186	0.209	0.180
	bl–ukiyo-e	0.247	0.231	0.189	0.235	0.208
FOG	baseline	**0.255**	**0.293**	**0.190**	0.252	**0.265**
	gen–autumn	0.219	0.236	0.155	0.217	0.213
	gen–sunset	0.171	0.223	0.126	0.178	0.178
	gen–ukiyo-e	0.232	0.250	0.173	**0.257**	0.228
	bl–autumn	0.194	0.218	0.113	0.197	0.179
	bl–sunset	0.187	0.220	0.099	0.198	0.184
	bl–ukiyo-e	0.197	0.225	0.128	0.219	0.193
SPRING	baseline	**0.246**	**0.254**	**0.200**	**0.265**	0.252
	gen–autumn	0.221	0.219	0.196	0.242	0.248
	gen–sunset	0.221	0.226	0.165	0.226	0.242
	bl–autumn	0.218	0.232	0.182	0.242	**0.256**
	bl–sunset	0.206	0.235	0.177	0.224	0.240
	bl–ukiyo-e	0.210	0.236	0.181	0.238	0.247

Table 3.1: *Εκπαίδευση του βασικού μοντέλου στο πεδίο New York City*

Old European City						
Train Domain	Prompt	DAWN	FOG	NIGHT	SPRING	WINTER
DAWN	baseline	**0.281**	**0.329**	**0.304**	**0.346**	**0.269**
	gen–autumn	0.226	0.264	0.247	0.266	0.228
	gen–sunset	0.240	0.270	0.232	0.266	0.216
	gen–ukiyo-e	0.221	0.263	0.238	0.265	0.226
	bl–autumn	0.186	0.209	0.192	0.221	0.175
	bl–sunset	0.193	0.218	0.200	0.215	0.166
	bl–ukiyo-e	0.210	0.248	0.229	0.250	0.205
FOG	baseline	**0.244**	**0.331**	**0.224**	**0.290**	**0.264**
	gen–autumn	0.224	0.272	0.201	0.262	0.219
	gen–sunset	0.197	0.231	0.162	0.181	0.173
	gen–ukiyo-e	0.217	0.281	0.218	0.282	0.222
	bl–autumn	0.193	0.227	0.165	0.211	0.170
	bl–sunset	0.206	0.229	0.157	0.212	0.174
	bl–ukiyo-e	0.196	0.231	0.193	0.237	0.183
SPRING	baseline	**0.286**	**0.289**	**0.263**	**0.309**	**0.266**
	gen–autumn	0.244	0.247	0.230	0.258	0.242
	gen–sunset	0.266	0.258	0.229	0.262	0.265
	bl–autumn	0.225	0.216	0.208	0.241	0.208
	bl–sunset	0.209	0.216	0.191	0.237	0.204
	bl–ukiyo-e	0.227	0.223	0.223	0.251	0.221

Table 3.2: *Εκπαίδευση του βασικού μοντέλου στο πεδίο Old European Town*

New York City (2)						
Train Domain	Prompt	DAWN	FOG	NIGHT	SPRING	WINTER
DAWN	baseline	**0.374**	**0.303**	**0.318**	**0.386**	**0.282**
	gen–autumn	0.328	0.277	0.263	0.304	0.238
	gen–sunset	0.323	0.251	0.216	0.295	0.197
	gen–ukiyo-e	0.337	0.263	0.264	0.324	0.231
	bl–autumn	0.297	0.240	0.191	0.274	0.200
	bl–sunset	0.303	0.248	0.215	0.269	0.183
	bl–ukiyo-e	0.312	0.267	0.231	0.287	0.207
FOG	baseline	**0.316**	**0.358**	**0.257**	0.358	**0.289**
	gen–autumn	0.264	0.308	0.220	0.335	0.255
	gen–sunset	0.252	0.297	0.201	0.303	0.211
	gen–ukiyo-e	0.296	0.320	0.244	**0.359**	0.256
	bl–autumn	0.254	0.299	0.197	0.316	0.231
	bl–sunset	0.252	0.292	0.197	0.300	0.222
	bl–ukiyo-e	0.244	0.290	0.211	0.307	0.229
SPRING	baseline	**0.325**	**0.314**	**0.311**	**0.381**	0.299
	gen–autumn	0.305	0.301	0.295	0.349	**0.303**
	gen–sunset	0.309	0.268	0.250	0.371	0.284
	bl–autumn	0.285	0.293	0.276	0.347	0.287
	bl–sunset	0.281	0.292	0.269	0.314	0.264
	bl–ukiyo-e	0.289	0.300	0.290	0.325	0.281

Table 3.3: *Εκπαίδευση του βασικού μοντέλου στο πεδίο New York City (2)*

Highway (2)						
Train Domain	Prompt	DAWN	FOG	NIGHT	SPRING	WINTER
DAWN	baseline	**0.415**	0.347	**0.356**	**0.389**	**0.386**
	gen–autumn	0.398	**0.388**	0.355	0.355	0.369
	gen–sunset	0.402	0.309	0.283	0.341	0.317
	gen–ukiyo-e	0.402	0.330	0.337	0.344	0.359
	bl–autumn	0.384	0.365	0.277	0.347	0.339
	bl–sunset	0.388	0.357	0.285	0.350	0.319
	bl–ukiyo-e	0.396	0.384	0.329	0.346	0.347
FOG	baseline	0.332	**0.424**	0.305	0.367	**0.388**
	gen–autumn	0.290	0.389	**0.389**	0.337	0.358
	gen–sunset	0.293	0.372	0.260	0.289	0.332
	gen–ukiyo-e	**0.363**	0.398	0.287	**0.387**	0.370
	bl–autumn	0.330	0.367	0.261	0.382	0.337
	bl–sunset	0.327	0.365	0.272	0.373	0.316
	bl–ukiyo-e	0.322	0.365	0.291	**0.387**	0.333
SPRING	baseline	**0.385**	**0.374**	**0.372**	**0.451**	**0.392**
	gen–autumn	0.369	0.357	0.368	0.429	0.365
	gen–sunset	0.383	0.319	0.313	0.423	0.358
	bl–autumn	0.324	0.325	0.347	0.417	0.343
	bl–sunset	0.352	0.357	0.314	0.424	0.344
	bl–ukiyo-e	0.322	0.365	0.291	0.387	0.333

Table 3.4: *Εκπαίδευση του βασικού μοντέλου στο πεδίο Highway (2)*

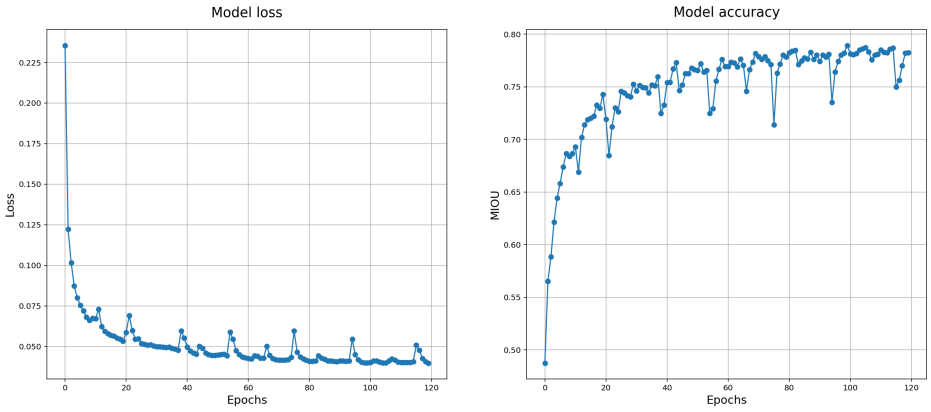
Παρατηρούμε ότι τα σύνολα δεδομένων Generated έχουν καλύτερη επίδοση από τα αντίστοιχα Blended. Εκτός από κάποιες περιπτώσεις, το μοντέλο που εκπαιδεύτηκε στα αρχικά

δεδομένα έχει την καλύτερη επίδοση. Φαίνεται ότι κάποια βελτίωση υπάρχει στη συνθήκη εκπαίδευσης FOG, για περιβάλλον εκπαίδευσης HIGHWAY (2), αλλά δεν μπορεί να εξαχθεί με βεβαιότητα κάποιο συμπέρασμα.

3.2 Αποτελέσματα από την εκπαίδευση με το επαυξημένο μοντέλο

Χρησιμοποιώντας τις ίδιες ονομασίες για την στήλη 'Mode', παρουσιάζουμε την επίδοση του μοντέλου με το τμήμα Αντικρουόμενης Μάθησης, ελέγχοντας με διάφορες προτρεπτικές λέξεις. Όπως πριν, συγκρίνουμε τα επαυξημένα σύνολα με το αρχικό ('augment'), και με την επίδοση του βασικού μοντέλου, εκπαιδευμένου στα αρχικά δεδομένα ('baseline').

Ένα παράδειγμα απώλειας και επίδοσης της εκπαίδευσης παρουσιάζεται στην εικόνα 3.2. Η επίδοση κάθε επιλογής παρουσιάζεται στους πίνακες 3.5, 3.6, 3.7, 3.8.



Εικόνα 3.2: *Διάγραμμα της απώλειας και της μετρικής MIOU στην εκπαίδευση*

New York City						
Train Domain	Mode	DAWN	FOG	NIGHT	SPRING	WINTER
DAWN	baseline	**0.305**	**0.282**	**0.249**	**0.288**	**0.268**
	augment	0.228	0.220	0.176	0.196	0.185
	gen–autumn	0.190	0.182	0.158	0.169	0.174
	gen–snowy	0.190	0.203	0.160	0.202	0.195
	gen–sunset	0.228	0.220	0.179	0.208	0.189
	bl–autumn	0.216	0.191	0.174	0.186	0.167
	bl–snowy	0.222	0.206	0.182	0.210	0.192
	bl–sunset	0.216	0.208	0.168	0.193	0.178
FOG	baseline	**0.255**	**0.293**	**0.190**	**0.252**	**0.265**
	augment	0.186	0.224	0.131	0.188	0.176
	gen–autumn	0.210	0.239	0.158	0.224	0.194
	gen–snowy	0.148	0.191	0.103	0.171	0.168
	gen–aurora	0.129	0.191	0.085	0.167	0.167
	bl–autumn	0.164	0.214	0.107	0.193	0.164
	bl–snowy	0.142	0.197	0.099	0.183	0.161
	bl–aurora	0.158	0.218	0.097	0.205	0.187
SPRING	baseline	**0.246**	**0.254**	**0.200**	**0.265**	0.252
	augment	0.227	0.249	0.190	0.262	**0.255**
	gen–autumn	0.211	0.227	0.186	0.226	0.242
	gen–snowy	0.152	0.194	0.143	0.172	0.200
	gen–aurora	0.162	0.205	0.141	0.202	0.227
	bl–autumn	0.180	0.229	0.132	0.230	0.239
	bl–snowy	0.135	0.193	0.111	0.193	0.210
	bl–aurora	0.170	0.211	0.145	0.213	0.229

Table 3.5: *Εκπαίδευση του επαυξημένου μοντέλου στο πεδίο New York City*

Old European City						
Train Domain	Mode	DAWN	FOG	NIGHT	SPRING	WINTER
DAWN	baseline	**0.281**	**0.329**	**0.304**	**0.346**	**0.269**
	augment	0.237	0.269	0.234	0.246	0.205
	gen–autumn	0.162	0.192	0.171	0.189	0.178
	gen–snowy	0.193	0.212	0.187	0.211	0.202
	gen–sunset	0.195	0.223	0.201	0.223	0.205
	bl–autumn	0.166	0.205	0.182	0.199	0.177
	bl–snowy	0.180	0.206	0.188	0.214	0.195
	bl–sunset	0.168	0.196	0.175	0.193	0.172
FOG	baseline	**0.244**	**0.331**	**0.224**	**0.290**	**0.264**
	augment	0.227	0.242	0.190	0.230	0.193
	gen–autumn	0.224	0.231	0.188	0.213	0.193
	gen–snowy	0.182	0.179	0.148	0.184	0.190
	gen–aurora	0.191	0.194	0.164	0.177	0.163
	bl–autumn	0.206	0.201	0.161	0.181	0.144
	bl–snowy	0.188	0.187	0.147	0.185	0.173
	bl–aurora	0.197	0.185	0.152	0.178	0.160
SPRING	baseline	**0.286**	**0.289**	**0.263**	**0.309**	0.266
	augment	0.262	0.257	0.226	0.267	**0.272**
	gen–autumn	0.242	0.238	0.219	0.241	0.249
	gen–snowy	0.179	0.170	0.148	0.182	0.200
	gen–aurora	0.208	0.195	0.177	0.198	0.208
	bl–autumn	0.223	0.191	0.169	0.203	0.212
	bl–snowy	0.181	0.148	0.130	0.175	0.187
	bl–aurora	0.191	0.172	0.154	0.182	0.188

Table 3.6: *Εκπαίδευση του επαυξημένου μοντέλου στο πεδίο Old European Town*

New York City (2)						
Train Domain	Mode	DAWN	FOG	NIGHT	SPRING	WINTER
DAWN	baseline	**0.374**	**0.303**	**0.318**	**0.386**	**0.282**
	augment	0.318	0.272	0.242	0.300	0.198
	gen–autumn	0.270	0.228	0.210	0.262	0.187
	gen–snowy	0.282	0.252	0.246	0.282	0.213
	gen–sunset	0.297	0.253	0.214	0.278	0.199
	bl–autumn	0.287	0.234	0.215	0.249	0.188
	bl–snowy	0.282	0.242	0.223	0.262	0.193
	bl–sunset	0.271	0.229	0.197	0.251	0.183
FOG	baseline	**0.316**	**0.358**	0.257	**0.358**	**0.289**
	augment	0.282	0.312	**0.278**	0.355	0.229
	gen–autumn	0.282	0.299	0.254	0.325	0.239
	gen–snowy	0.232	0.273	0.247	0.302	0.230
	gen–aurora	0.204	0.253	0.211	0.255	0.210
	bl–autumn	0.230	0.279	0.214	0.301	0.218
	bl–snowy	0.229	0.290	0.246	0.298	0.227
	bl–aurora	0.230	0.287	0.242	0.315	0.233
SPRING	baseline	**0.325**	0.314	**0.311**	0.381	0.299
	augment	0.304	**0.330**	0.294	**0.385**	0.313
	gen–autumn	0.306	0.314	0.300	0.349	**0.330**
	gen–snowy	0.250	0.274	0.276	0.302	0.269
	gen–aurora	0.257	0.279	0.268	0.324	0.275
	bl–autumn	0.254	0.284	0.266	0.340	0.307
	bl–snowy	0.217	0.266	0.257	0.295	0.271
	bl–aurora	0.245	0.277	0.258	0.304	0.273

Table 3.7: *Εκπαίδευση του επαυξημένου μοντέλου στο πεδίο New York City (2)*

Highway (2)						
Train Domain	Mode	DAWN	FOG	NIGHT	SPRING	WINTER
DAWN	baseline	**0.415**	0.347	**0.356**	**0.389**	**0.386**
	augment	0.408	**0.379**	0.323	0.354	0.346
	gen–autumn	0.366	0.351	0.308	0.316	0.327
	gen–snowy	0.385	0.371	0.332	0.341	0.343
	gen–sunset	0.386	0.366	0.285	0.325	0.335
	bl–autumn	0.369	0.365	0.308	0.328	0.326
	bl–snowy	0.373	0.362	0.305	0.320	0.319
	bl–sunset	0.360	0.353	0.285	0.328	0.316
FOG	baseline	0.332	**0.424**	0.305	0.367	**0.388**
	augment	**0.374**	0.382	0.334	**0.411**	0.361
	gen–autumn	0.364	0.358	0.326	0.397	0.352
	gen–snowy	0.346	0.363	**0.341**	0.403	0.351
	gen–aurora	0.289	0.313	0.313	0.358	0.282
	bl–autumn	0.304	0.320	0.300	0.365	0.288
	bl–snowy	0.316	0.333	0.340	0.382	0.316
	bl–aurora	0.305	0.319	0.334	0.376	0.287
SPRING	baseline	**0.385**	0.374	0.372	**0.451**	**0.392**
	augment	0.350	**0.376**	0.352	0.426	0.363
	gen–autumn	0.374	0.371	**0.384**	0.421	0.370
	gen–snowy	0.351	0.351	0.359	0.416	0.355
	gen–aurora	0.330	0.332	0.333	0.399	0.324
	bl–autumn	0.321	0.327	0.345	0.403	0.344
	bl–snowy	0.307	0.310	0.353	0.403	0.331
	bl–aurora	0.325	0.336	0.343	0.405	0.339

Table 3.8: *Εκπαίδευση του επαυξημένου μοντέλου στο πεδίο Highway (2)*

Και σε αυτή την περίπτωση το baseline είχε την καλύτερη επίδοση. Το επαυξημένο μοντέλο, εκπαιδευμένο στο αρχικό σύνολο δεδομένων είχε καλύτερη επίδοση σε σχέση με τα μοντέλα που εκπαιδεύτηκαν στα επαυξημένα σύνολα δεδομένων. Στο πεδίο που είναι πιο κοντά στο πεδίο εκπαίδευσης (HIGHWAY (2)) φαίνεται ότι υπάρχει μια βελτίωση στις συνθήκες που είναι πιο μακριά από την συνθήκη εκπαίδευσης.

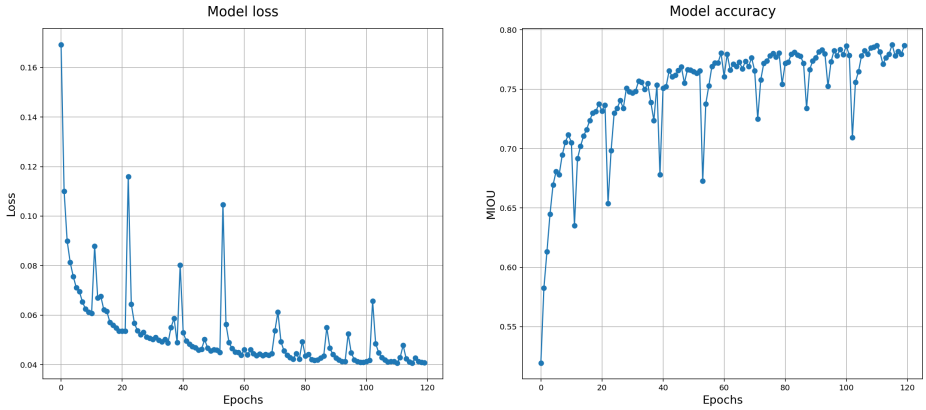
3.3 Αποτελέσματα από επιπρόσθετες συνθήκες εκπαίδευσης του επαυξημένου μοντέλου

Στο προηγούμενο μέρος χρησιμοποιήσαμε στην εκπαίδευση την αρχιτεκτονική [30]. Σε αυτό το μέρος εφαρμόζουμε διάφορες εναλλακτικές στην επαυξημένη αρχιτεκτονική και συγκρίνουμε τα αποτελέσματα:

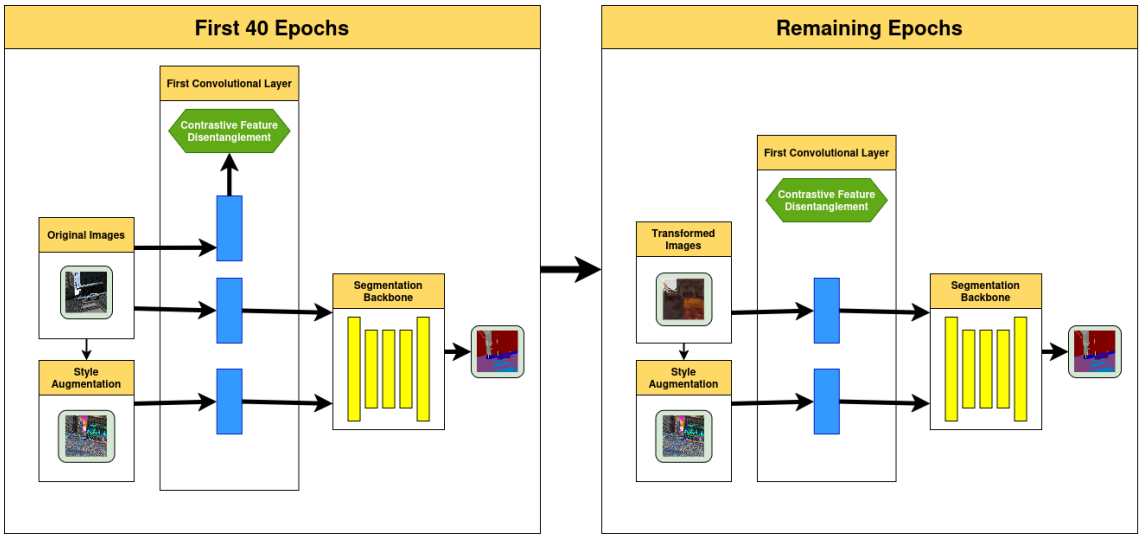
- Επιλογή **A**: Για τις πρώτες 40 εποχές εκπαίδευσης το αρχικό σύνολο δεδομένων τροφοδοτείται στην επαυξημένη αρχιτεκτονική. Για τον υπόλοιπο αριθμό εποχών τροποποιούμε το μοντέλο ώστε να σταματά την διαδικασία Αντικρουόμενης Μάθησης και εκπαιδεύουμε χρησιμοποιόντας το επαυξημένο σύνολο δεδομένων.
- Επιλογή **B**: Για κάθε εποχή εκπαιδεύουμε το τμήμα Αντικρουόμενης Μάθησης στο αρχικό σύνολο δεδομένων και στη συνέχεια εκπαιδεύουμε το μοντέλο στο επαυξημένο σύνολο δεδομένων.
- Επιλογή **C**: Για κάθε εποχή εκπαιδεύουμε το τμήμα Αντικρουόμενης Μάθησης και εφαρμόζουμε τις επαυξήσεις στυλ στο αρχικό σύνολο δεδομένων, και στη συνέχεια εκπαιδεύουμε με το επαυξημένο σύνολο δεδομένων
- Επιλογή **D**: Για τις πρώτες 40 εποχές εκπαιδεύουμε με το αρχικό σύνολο δεδομένων, και στη συνέχεια εκπαιδεύουμε με το επαυξημένο σύνολο δεδομένων.

Δοκιμάστηκαν αρκετές ακόμα συνθήκες εκπαίδευσης, με την επίδοσή τους να είναι σημαντικά χειρότερη από το baseline. Παρακάτω δείχνουμε τους συνδυασμούς που έδειξαν κάποια βελτίωση σε ένα ή περισσότερα πεδία. Τα μοντέλα συγκρίνονται με το baseline και το augment, όπως και πριν. Ο τίτλος κάθε πίνακα περιγράφει ποιά έξοδο του μοντέλου DiffuseMix χρησιμοποιείται, καθώς και την προτρεπτική λέξη και το πεδίο εκπαίδευσης.

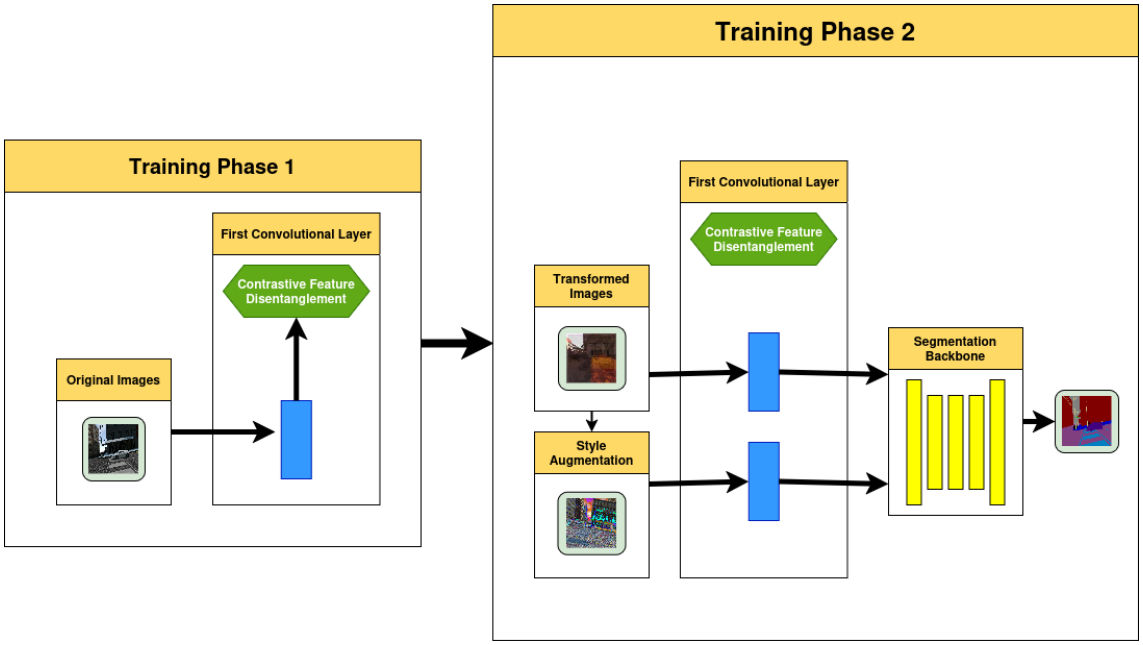
Ένα παράδειγμα απώλειας και επίδοσης της εκπαίδευσης παρουσιάζεται στην εικόνα 3.3. Οι συνθήκες που εξετάζουμε απεικονίζονται στα διαγράμματα 3.4, 3.5, 3.6, 3.7. Η επίδοση κάθε επιλογής παρουσιάζεται στους πίνακες 3.9, 3.10, 3.11, 3.12.



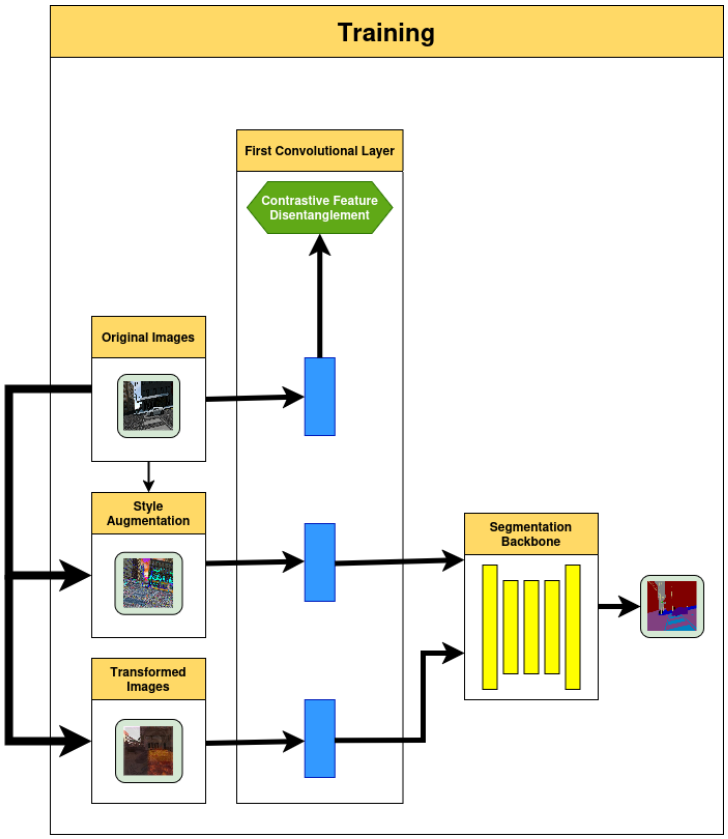
Εικόνα 3.3: *Διάγραμμα της απώλειας και της μετρικής MIOU στην εκπαίδευση*



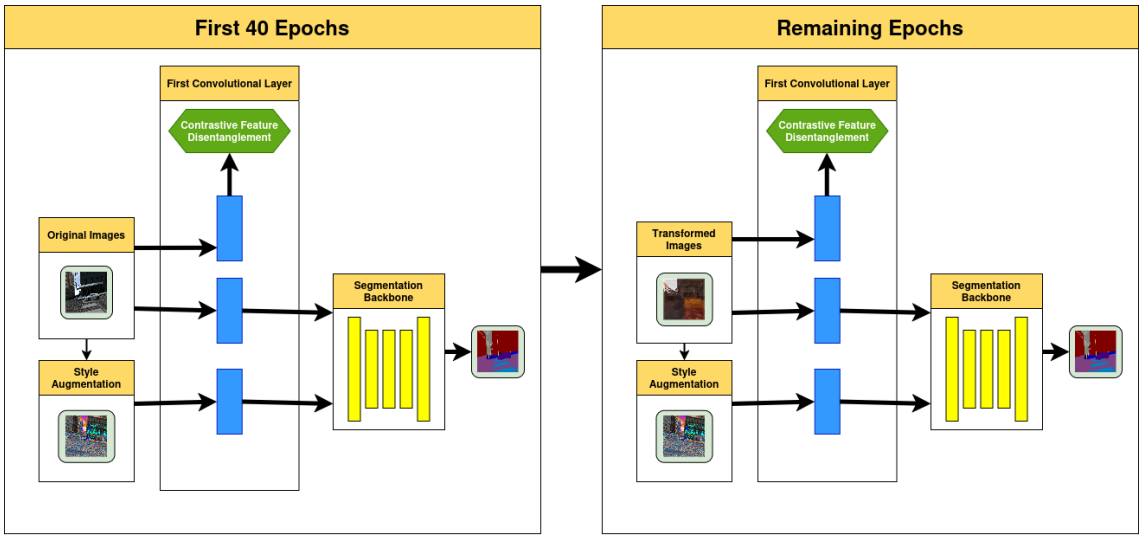
Εικόνα 3.4: *Διάγραμμα της επιλογής A*



Εικόνα 3.5: *Διάγραμμα της επιλογής B*



Εικόνα 3.6: *Διάγραμμα της επιλογής C*



Εικόνα 3.7: *Διάγραμμα της επιλογής D*

generated,snowy,fog						
Test Domain	Mode	DAWN	FOG	NIGHT	SPRING	WINTER
New York City	baseline	**0.255**	**0.293**	**0.190**	**0.252**	**0.265**
	augment	0.186	0.224	0.131	0.188	0.176
	A	0.198	0.246	0.170	0.215	0.216
	B	0.159	0.217	0.129	0.180	0.176
	C	0.186	0.223	0.141	0.202	0.192
	D	0.154	0.206	0.112	0.171	0.169
Old European Town	baseline	**0.244**	**0.331**	**0.224**	**0.290**	**0.264**
	augment	0.227	0.242	0.190	0.230	0.193
	A	0.232	0.261	0.222	0.257	0.248
	B	0.188	0.199	0.163	0.201	0.210
	C	0.239	0.251	0.209	0.243	0.222
	D	0.182	0.186	0.147	0.180	0.187
New York City (2)	baseline	**0.316**	**0.358**	0.257	**0.358**	**0.289**
	augment	0.282	0.312	0.278	0.355	0.229
	A	0.296	0.336	**0.303**	0.350	0.278
	B	0.255	0.300	0.265	0.315	0.245
	C	0.264	0.313	0.272	0.336	0.242
	D	0.241	0.286	0.252	0.310	0.228
Highway (2)	baseline	0.332	**0.424**	0.305	0.367	**0.388**
	augment	**0.374**	0.382	0.334	0.411	0.361
	A	0.368	0.382	**0.368**	**0.414**	0.366
	B	0.357	0.370	0.353	0.405	0.360
	C	0.372	0.387	0.345	0.412	0.372
	D	0.344	0.369	0.340	0.400	0.357

Table 3.9: *'Ελεγχος προσεγγίσεων fine-tuning στο επαυξημένο μουτέλο με generated εικόνες με την προτρεπτική λέξη 'snowy'. Η εκπαίδευση έγινε στο περιβάλλον FOG*

blended,snowy,dawn						
Test Domain	Mode	DAWN	FOG	NIGHT	SPRING	WINTER
New York City	baseline	**0.305**	**0.282**	**0.249**	**0.288**	**0.268**
	augment	0.228	0.220	0.176	0.196	0.185
	A	0.225	0.232	0.197	0.209	0.194
	B	0.222	0.193	0.180	0.200	0.179
	C	0.223	0.205	0.171	0.192	0.172
	D	0.211	0.187	0.179	0.190	0.169
Old European Town	baseline	**0.281**	**0.329**	**0.304**	**0.346**	**0.269**
	augment	0.237	0.269	0.234	0.246	0.205
	A	0.211	0.242	0.217	0.238	0.211
	B	0.175	0.193	0.175	0.204	0.190
	C	0.188	0.215	0.189	0.210	0.177
	D	0.169	0.187	0.170	0.197	0.178
New York City (2)	baseline	**0.374**	**0.303**	**0.318**	**0.386**	**0.282**
	augment	0.318	0.272	0.242	0.300	0.198
	A	0.297	0.269	0.253	0.287	0.203
	B	0.266	0.233	0.223	0.268	0.190
	C	0.287	0.239	0.221	0.254	0.177
	D	0.258	0.224	0.224	0.250	0.183
Highway (2)	baseline	**0.415**	0.347	**0.356**	**0.389**	**0.386**
	augment	0.408	0.379	0.323	0.354	0.346
	A	0.390	**0.384**	0.328	0.350	0.345
	B	0.360	0.360	0.304	0.326	0.310
	C	0.389	0.380	0.326	0.336	0.337
	D	0.370	0.368	0.302	0.333	0.316

Table 3.10: *'Ελεγχος προσεγγίσεων fine-tuning στο επαυξημένο μοντέλο με generated εικόνες με την προτρεπτική λέξη 'snowy'. Η εκπαίδευση έγινε στο περιβάλλον DAWN*

ukiyo-e,fog						
Test Domain	Mode	DAWN	FOG	NIGHT	SPRING	WINTER
New York City	baseline	**0.255**	**0.293**	**0.190**	**0.252**	**0.265**
	augment	0.186	0.224	0.131	0.188	0.176
	bl−A	0.174	0.237	0.129	0.204	0.189
	gen−A	0.202	0.234	0.161	0.219	0.197
	bl−D	0.192	0.226	0.148	0.205	0.179
	gen−D	0.199	0.236	0.153	0.213	0.211
Old European Town	baseline	**0.244**	**0.331**	**0.224**	**0.290**	**0.264**
	augment	0.227	0.242	0.190	0.230	0.193
	bl−A	0.195	0.206	0.172	0.203	0.180
	gen−A	0.227	0.240	0.210	0.249	0.211
	bl−D	0.207	0.218	0.179	0.215	0.180
	gen−D	0.240	0.255	0.222	0.256	0.219
New York City (2)	baseline	**0.316**	**0.358**	0.257	**0.358**	**0.289**
	augment	0.282	0.312	0.278	0.355	0.229
	bl−A	0.248	0.303	0.265	0.314	0.236
	gen−A	0.277	0.307	**0.294**	0.351	0.243
	bl−D	0.252	0.303	0.267	0.320	0.230
	gen−D	0.283	0.308	0.285	0.347	0.247
Highway(2)	baseline	0.332	**0.424**	0.305	0.367	**0.388**
	augment	**0.374**	0.382	0.334	0.411	0.361
	bl−A	0.341	0.359	0.350	0.394	0.341
	gen−A	0.372	0.376	0.363	0.410	0.370
	bl−D	0.331	0.357	0.361	0.390	0.334
	gen−D	0.369	0.388	**0.369**	**0.414**	0.370

Table 3.11: *Έλεγχος προσεγγίσεων A και D στο επαυξημένο μοντέλο με generated και blended εικόνες με την προτρεπτική λέξη 'ukiyo-e'. Η εκπαίδευση έγινε στο περιβάλλον FOG*

ukiyo-e,spring						
Test Domain	Mode	DAWN	FOG	NIGHT	SPRING	WINTER
New York City	baseline	**0.246**	**0.254**	**0.200**	**0.265**	0.252
	augment	0.227	0.249	0.190	0.262	**0.255**
	bl−A	0.197	0.218	0.169	0.224	0.237
	gen−A	0.206	0.219	0.186	0.218	0.238
	bl−D	0.195	0.226	0.163	0.234	0.243
	gen−D	0.203	0.233	0.184	0.214	0.239
Old European Town	baseline	**0.286**	**0.289**	**0.263**	**0.309**	0.266
	augment	0.262	0.257	0.226	0.267	**0.272**
	bl−A	0.227	0.198	0.186	0.218	0.213
	gen−A	0.236	0.222	0.214	0.231	0.240
	bl−D	0.236	0.196	0.178	0.211	0.218
	gen−D	0.246	0.228	0.215	0.240	0.245
New York City (2)	baseline	**0.325**	0.314	**0.311**	0.381	0.299
	augment	0.304	**0.330**	0.294	**0.385**	**0.313**
	bl−A	0.273	0.284	0.284	0.326	0.285
	gen−A	0.301	0.301	0.304	0.351	0.306
	bl−D	0.276	0.290	0.286	0.341	0.291
	gen−D	0.297	0.294	0.300	0.343	0.292
Highway (2)	baseline	**0.385**	0.374	0.372	**0.451**	**0.392**
	augment	0.350	**0.376**	0.352	0.426	0.363
	bl−A	0.346	0.347	0.380	0.417	0.346
	gen−A	0.366	0.365	0.384	0.418	0.367
	bl−D	0.339	0.341	0.370	0.405	0.346
	gen−D	0.372	0.364	**0.394**	0.420	0.364

Table 3.12: *Έλεγχος προσεγγίσεων A και D στο επαυξημένο μουτέλο με generated και blended εικόνες με την προτρεπτική λέξη 'ukiyo-e'. Η εκπαίδευση έγινε στο περιβάλλον SPRING*

Παρατηρούμε ότι και σε αυτή την περίπτωση οι αλλαγές που κάναμε δεν επέφεραν κάποια σημαντική βελτίωση, με εξαίρεση ξανά το περιβάλλον Highway (2) στην συνθήκη NIGHT. Παρατηρήθηκε επίσης βελτιωμένη επίδοση των εικόνων που δημιουργήθηκαν από το μοντέλο διάχυσης σε σύγκριση με τις εικόνες που έχουν αναμειχθεί με φράκταλ.

3.4 Περαιτέρω διερεύνηση του περιβάλλοντος Highway (2)

Παρατηρήσαμε προηγουμένως ότι σε σύγκριση με τα άλλα περιβάλλοντα, το περιβάλλον Highway (2) είχε καλύτερες επιδόσεις. Παρακάτω παρουσιάζουμε τις επιδόσεις εκπαίδευσης σε αυτό το περιβάλλον, με στόχο να βρούμε κάποιο μοτίβο.

Η επίδοση κάθε επιλογής παρουσιάζεται στους πίνακες 3.13, 3.14, 3.15.

Train Domain: Dawn						
Mode	Prompt	DAWN	FOG	NIGHT	SPRING	WINTER
baseline	-	**0.415**	0.347	**0.356**	**0.389**	**0.386**
augment	-	0.408	0.379	0.323	0.354	0.346
base-bl	autumn	0.384	0.365	0.277	0.347	0.339
base-gen	autumn	0.398	0.388	0.355	0.355	0.369
base-bl	ukiyo-e	0.396	0.384	0.329	0.346	0.347
base-gen	ukiyo-e	0.402	0.330	0.337	0.344	0.359
augment-bl	autumn	0.369	0.365	0.308	0.328	0.326
augment-gen	autumn	0.366	0.351	0.308	0.316	0.327
augment-bl	snowy	0.373	0.362	0.305	0.320	0.319
augment-gen	snowy	0.385	0.371	0.332	0.341	0.343
bl-A	autumn	0.385	0.374	0.336	0.350	0.339
gen-A	autumn	0.395	0.379	0.342	0.348	0.354
bl-A	snowy	0.390	0.384	0.328	0.350	0.345
gen-A	snowy	0.387	0.376	0.342	0.348	0.343
bl-A	ukiyo-e	0.388	0.378	0.316	0.323	0.340
gen-A	ukiyo-e	0.392	0.363	0.333	0.332	0.349
bl-B	autumn	0.360	0.349	0.291	0.318	0.331
gen-B	autumn	0.386	0.379	0.337	0.337	0.350
bl-B	snowy	0.360	0.360	0.304	0.326	0.310
gen-B	snowy	0.402	0.374	0.329	0.347	0.356
bl-C	autumn	0.397	0.377	0.318	0.342	0.349
bl-C	ukiyo-e	0.412	**0.394**	0.340	0.356	0.347
bl-D	snowy	0.370	0.368	0.302	0.333	0.316
gen-D	snowy	0.386	0.378	0.342	0.341	0.341
bl-D	ukiyo-e	0.398	0.389	0.342	0.351	0.349
gen-D	ukiyo-e	0.394	0.372	0.346	0.341	0.345

Table 3.13: *Επίδοση του μοντέλου στο περιβάλλον Highway (2) με συνθήκη εκπαίδευσης DAWN*

Train Domain: Fog						
Mode	Prompt	DAWN	FOG	NIGHT	SPRING	WINTER
baseline	-	0.332	**0.424**	0.305	0.367	**0.388**
augment	-	**0.374**	0.382	0.334	0.411	0.361
base-bl	autumn	0.330	0.367	0.261	0.382	0.337
base-gen	autumn	0.290	0.389	0.271	0.337	0.358
base-bl	ukiyo-e	0.322	0.365	0.291	0.387	0.333
base-gen	ukiyo-e	0.363	0.398	0.287	0.387	0.370
augment-bl	autumn	0.304	0.320	0.300	0.365	0.288
augment-gen	autumn	0.364	0.358	0.326	0.397	0.352
augment-bl	snowy	0.316	0.333	0.340	0.382	0.316
augment-gen	snowy	0.346	0.363	0.341	0.403	0.351
bl-A	aurora	0.338	0.360	0.345	0.391	0.331
gen-A	aurora	0.338	0.362	0.330	0.387	0.336
bl-A	snowy	0.346	0.366	0.358	0.403	0.350
gen-A	snowy	0.368	0.382	0.368	**0.414**	0.366
bl-A	ukiyo-e	0.341	0.359	0.350	0.394	0.341
gen-A	ukiyo-e	0.372	0.376	0.363	0.410	0.370
gen-B	snowy	0.357	0.370	0.353	0.405	0.360
gen-B	ukiyo-e	0.363	0.385	0.365	0.411	0.368
gen-C	snowy	0.372	0.387	0.345	0.412	0.372
bl-D	snowy	0.316	0.331	0.345	0.390	0.326
gen-D	snowy	0.344	0.369	0.340	0.400	0.357
bl-D	ukiyo-e	0.331	0.357	0.361	0.390	0.334
gen-D	ukiyo-e	0.369	0.388	**0.369**	**0.414**	0.370

Table 3.14: *Επίδοση του μοντέλου στο περιβάλλον Highway (2) με συνθήκη εκπαίδευσης FOG*

Train Domain: Spring						
Mode	Prompt	DAWN	FOG	NIGHT	SPRING	WINTER
baseline	-	**0.385**	0.374	0.372	**0.451**	**0.392**
augment	-	0.350	**0.376**	0.352	0.426	0.363
base-bl	autumn	0.324	0.325	0.347	0.417	0.343
base-gen	autumn	0.369	0.357	0.368	0.429	0.365
augment-bl	aurora	0.325	0.336	0.343	0.405	0.339
augment-gen	aurora	0.330	0.332	0.333	0.399	0.324
augment-bl	autumn	0.321	0.327	0.345	0.403	0.344
augment-gen	autumn	0.374	0.371	0.384	0.421	0.370
augment-bl	snowy	0.307	0.310	0.353	0.403	0.331
augment-gen	snowy	0.351	0.351	0.359	0.416	0.355
bl-A	aurora	0.357	0.362	0.363	0.424	0.361
gen-A	aurora	0.371	0.365	0.355	0.420	0.367
bl-A	autumn	0.374	0.370	0.384	0.428	0.372
gen-A	autumn	0.383	0.372	**0.398**	0.435	0.383
bl-A	ukiyo-e	0.346	0.347	0.380	0.417	0.346
gen-A	ukiyo-e	0.366	0.365	0.384	0.418	0.367
bl-B	ukiyo-e	0.342	0.347	0.365	0.408	0.344
gen-C	snowy	0.367	0.369	0.372	0.435	0.366
bl-D	ukiyo-e	0.339	0.341	0.370	0.405	0.346
gen-D	ukiyo-e	0.372	0.364	0.394	0.420	0.364

Table 3.15: *Επίδοση του μοντέλου στο περιβάλλον Highway (2) με συνθήκη εκπαίδευσης SPRING*

Βλέπουμε πιο ξεκάθαρα ότι για όμοια περιβάλλοντα η μέθοδος μας μπορεί επιφέρει βελτίωση, ιδιαίτερα για τις συνθήκες που είναι πιο μακριά από τις συνθήκες εκπαίδευσης, όπως η FOG με την εκπαίδευση να έχει γίνει στο DAWN, ή η FOG και NIGHT με την εκπαίδευση να έχει γίνει στο SPRING. Η μέθοδος μας βελτιώνει την σημαντικά την επίδοση στην εκπαίδευση στο FOG, για τις συνθήκες ελέγχου DAWN και SPRING.

3.5 Συμπεράσματα και μελλοντικές επεκτάσεις

Σε αυτή την εργασία δείξαμε ότι το επαυξημένο μοντέλο που χρησιμοποιήσαμε μπορεί να επιφέρει βελτίωση ως προς την αρχική επίδοση.

Η βελτίωση στην επίδοση ήταν πιο ξεκάθαρη στο περιβάλλον που έχει τη μεγαλύτερη σχέση με το περιβάλλον εκπαίδευσης, και για τις συνθήκες με τη μέγιστη απόσταση από την συνθήκη εκπαίδευσης. Συμπεραίνουμε ότι η αρχιτεκτονική μας μπορεί να χρησιμοποιηθεί για την προσαρμογή ενός μοντέλου σε δύσκολα πεδία. Ακόμη, η χρήση προτρεπτικών λέξεων μπορεί να επηρεάσει την επίδοση του μοντέλου, και παρατηρήθηκαν μεγάλες αποκλίσεις με τη χρήση διαφορετικών λέξεων, εντός του ίδιου περιβάλλοντος. Για κάθε περιβάλλον και

συνθήκη υπήρχαν διαφορετικές προτρεπτικές λέξεις που σημείωναν τη βέλτιστη απόδοση, με τη σχέση μεταξύ τους να μην είναι πάντα προφανής.

Παρόλα αυτά, για τα περισσότερα περιβάλλοντα το μοντέλο δεν προσέφερε κάποια βελτίωση στην επίδοση. Αυτό μπορεί να αποδοθεί σε μη επαρκή κανονικοποίηση ή κάποιο άλλο πρόβλημα που παρά την διεξοδική αναζήτηση δεν επιλύθηκε. Μπορεί επίσης το συγκεκριμένο μοντέλο να μην έχει εν γένει την ικανότητα να ανταπεξέλθει σε αυτή την εργασία, σημειώνοντας το γεγονός ότι το μοντέλο CCSDG αναπτύχθηκε για μία εργασία δυαδικής σημασιολογικής κατάτμησης, επομένως μπορεί να συναντά δυσκολία στην κατάτμηση με πολλές κλάσεις του συνόλου δεδομένων SYNTHIA.

Υπάρχουν πολλά μονοπάτια που μπορούμε να διερευνήσουμε. Μία πρώτη ιδέα είναι να επεκτείνουμε τη χρήση του μοντέλου DiffuseMix, συνδυάζοντας στο ίδιο σύνολο δεδομένων εικόνες που παράγονται από διαφορετικές προτρεπτικές λέξεις. Η χρήση ενός άλλου μοντέλου, όπως του μοντέλου Mask R-CNN [3] που έχει σχεδιαστεί για εργασίες κατάτμησης περιστατικών, μπορεί να συνδυαστεί με το μοντέλο FCN που χρησιμοποιούμε. Η υπάρχουσα αρχιτεκτονική μπορεί να δοκιμαστεί σε μία από τις κλασσικές εργασίες Γενίκευσης Πεδίου, όπως η SYNTHIA→Cityscapes, για περαιτέρω έλεγχο του μοντέλου. Τέλος, μία σε βάθος ανάλυση της επίδοσης κάθε κλάσης ξεχωριστά, και ποιές μέθοδοι έχουν τη μεγαλύτερη επίδραση στην επίδοση μπορεί να οδηγήσει στην βελτίωση του μοντέλου.

Part II

English Version

Chapter 4

Introduction

Recent developments in Machine Learning have elevated Artificial Intelligence into a staple of modern life. Many production sectors are being transformed by the introduction of complex models capable of monitoring, deciding and compiling solutions for a variety of problems. AI has revolutionized the way knowledge is created and disseminated.

The field of Computer Vision, one of the most important fields of Artificial Intelligence, facilitate the creation of models that enable computer systems to interpret the visual world. Its core features, object detection and recognition, image classification, segmentation and creation, have found many applications in autonomous driving systems, manufacturing and healthcare.

4.1 Motivation

Machine Learning models require extensive training datasets to increase their performance and robustness. In Computer Vision, the relative difficulty of obtaining, processing and annotating training datasets, and the computationally demanding training process encouraged efforts to create models that retain their predictive capabilities beyond the data they were trained on, or for inventing techniques that enable already existing models to better adapt to new data. A new area of extensive study has emerged, named Domain Adaptation or Generalization, the difference being respectively in the possible knowledge of other domain distributions or not. Many Domain Adaptation techniques have been developed that enable models to better perform in unknown and adverse conditions, increasing the usefulness of these models in many tasks.

Domain Adaptation methods are crucial in the fields of medical imaging and autonomous driving. The differences between imaging equipment, the skill consistency of the operators and affordability limit greatly the availability of training medical data. In autonomous driving, the major hurdle is the annotation of collected image data, which is a time-consuming and thus expensive task. In addition, in both fields models with near zero fault tolerance and quick decision-making are demanded, especially in the case of autonomous driving, where no human can vet on-time the decision of the model. For these reasons, domain adaptation and generalization approaches are often aimed at improving these tasks. A solution to the limitations of training data availability is the utilization of synthetic data, such as the *SYNTHIA* [11] datasets of urban environments, with

the purpose to provide models with near-realistic training data that can be subsequently fine-tuned, using generalization methods, for prospective tasks in real urban datasets.

Domain Generalization techniques aim to bring to the surface underlying structural information from the training data, considered to be invariant in all environmental conditions. This information is learned by the model, which is tested across unfamiliar domains to evaluate its generalization capabilities. Data augmentation methods can alter the distribution of the image and are proven to facilitate the generalization performance of the accompanied models in certain settings. Contrastive and adversarial training, with the inclusion of a generative model, are widely studied and have improved many Domain Adaptation tasks.

Building on existing methods for domain generalization, we explore the generalization capabilities of the pipeline suggested by Hu et al. [30] in training environments provided by the SYNTHIA dataset. Several transformative methods are applied to the input data, including the leveraging of a diffusion model, controlled by textual prompts, to generate novel images for training.

4.2 Structure

The thesis is organized in 5 chapters.

In chapter 4 we introduce the basic ideas behind our study, and the motivation behind our choices.

Chapter 5 is dedicated to the theoretical background of the study, providing an understanding of the methods used. First, we describe the basic concepts of deep learning and their applications in neural networks. We emphasize on the structure of Convolutional Neural Networks and their augmentations in Residual Networks and Fully Convolutional Networks. An introduction to Diffusion Models is then provided. Finally, we outline the basic ideas and approaches behind Domain Adaptation and Generalization.

In chapter 6 we provide an overview of several related works and we outline the methodology and the details of the experiment. We describe approaches in Domain Adaptation and Generalization tasks, advancements in Diffusion Models and Data Augmentation methods. Then we present our method, the source dataset and the modifications we applied, our training setup and details of the implementation.

In chapter 7 we present the experimental settings and an review of their results. For each experiment we evaluate the result and the methods used.

Finally, in chapter 8 we provide an evaluation of our methods and the possible directions of a continuation of this work.

Chapter 5

Theoretical Background

In this chapter we outline the basic concepts behind the processes we used in this work. We begin with an introduction to the tasks of Image Segmentation. We then proceed with an in-depth overview of convolutional networks and their modifications in residual CNN and Fully CNN. We explore the ideas behind Diffusion Models. Finally we provide with an analysis of Domain Adaptation and Generalization and the multiple research directions of their applications.

5.1 Image Segmentation

Image Segmentation is a core computer vision task [15]. In general, it involves partitioning an image into multiple segments, each representing a different object. Along with object detection, image segmentation has applications in medical image analysis [16], autonomous vehicles [17], surveillance etc. Multiple examples of segmented images are presented in figure 5.1.

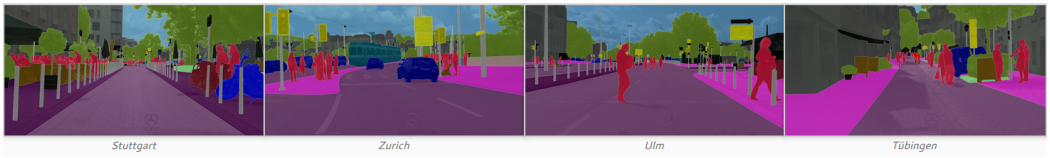


Image 5.1: *An overview of segmented images of urban environments by [1]*

Generally a deep neural network is trained on image data and learn to densely assign a label to each pixel. This classification task of separating regions of the original image with semantic labels is known as semantic segmentation (5.2). An additional partitioning of individual objects by detecting multiple instances of an object, on top of their semantic annotation, is known as instance segmentation (5.3). In general, segmentation tasks are significantly harder than simple classification tasks, especially considering their computational requirements.

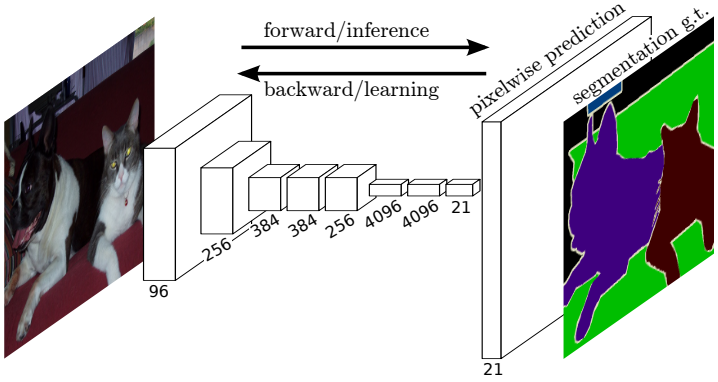


Image 5.2: *Semantic Segmentation example using a FCN [2]*

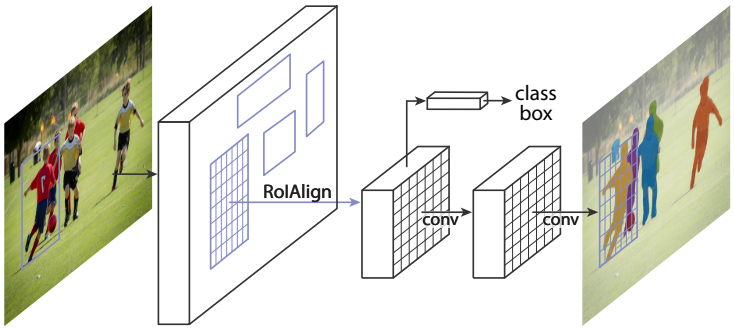


Image 5.3: *Instance Segmentation example with Mask R-CNN [3]*

5.2 Machine Learning

Machine Learning is a field of study in Artificial Intelligence that is concerned with the development of algorithms that can learn to perform tasks and generalize to unseen data without being given explicit instructions. Machine learning algorithms aim to learn patterns embedded in their training data, to make a prediction or a decision based on their accumulated knowledge.

Machine Learning algorithms are categorized with respect to the nature of their input. Traditionally these categories are:

- **Supervised Learning**
- **Unsupervised Learning**
- **Reinforcement Learning**

5.2.1 Supervised Learning

In supervised learning, the algorithm is trained on a set of data that contains the desired outputs. The input data consists of training examples which are represented by feature vectors x. Each training example has one or more inputs as the desired output y, called a supervisory signal. Training aims to find a mapping $y = f(x)$ from the input to output that can make accurate predictions or decisions on unseen data. Supervised Learning is typically used in Classification or Regression problems. In Classification

problems a model is trained to assign a label or a label vector to each training example. Labels are usually a set of predetermined set of categorical values, relevant to the training data. Depending on the size of the label space, classification problems are divided to binary (for 2 classes) and multiclass (for 3 or more classes). In Regression problems the model returns a numerical value estimation on a particular attribute of the input data. In general, classification problems return discrete values and regression problems return continuous values.

5.2.2 Unsupervised Learning

In unsupervised Learning the input data do not contain any type of labelling to guide the training process. The goal of training is to find structures or relations within the data. The two most common types of Unsupervised Learning algorithms are Clustering and Dimensionality Reduction. Clustering algorithms divide the input data into groups or clusters based on learned similarities. The K-means algorithm is categorized as an unsupervised clustering algorithm. Dimensionality reduction is the transformation of the data from a high dimensional space into a lower dimensional space, aiming to provide a more meaningful representation of the original data. The Principal Component Analysis (PCA) algorithm performs dimensionality reduction to obtain the principal dimensions.

Self-Supervised Learning is a type of unsupervised learning where a supervisory signal for the input data is generated by the algorithm. Semi-Supervised Learning combines supervised with unsupervised learning. The model is trained with labeled and unlabeled data, in order to improve the models performance.

5.2.3 Reinforcement Learning

In Reinforcement Learning an agent is trained to make sequential decisions, with the goal of maximizing a 'score' metric. A virtual environment is created, where the agent learns through trial and error, reacting to the conditions and changes of the environment. Reinforcement Learning algorithms are used in autonomous driving or in game AIs.

5.3 Neural Networks

5.3.1 Artificial Neural Networks

Artificial Neural Networks (ANN) are computational systems modeled after biological neuronal systems [6]. A number of interconnected nodes, referred to as neurons, are arranged in consecutive layers. Each layer performs a specified function, according to the architecture of the model. The basic structure of an ANN, as shown in the image 5.4, consists of an input layer, matching the input data dimensions, a number of hidden layers processing the input and an output layer that produces a desired outcome. Each neuron recieves input signals, which are weighted with a weighing function. The weighted sum is then used to make a decision, by passing through an activation function, that is transmitted to the connected neurons.

The learning process is done by adjusting the weight parameters of the neurons, using the backpropagation algorithm, with the aim to minimize a metric of difference between the predicted and the desired output, called a loss function. The selection of the activation and the loss functions are relevant to the task.

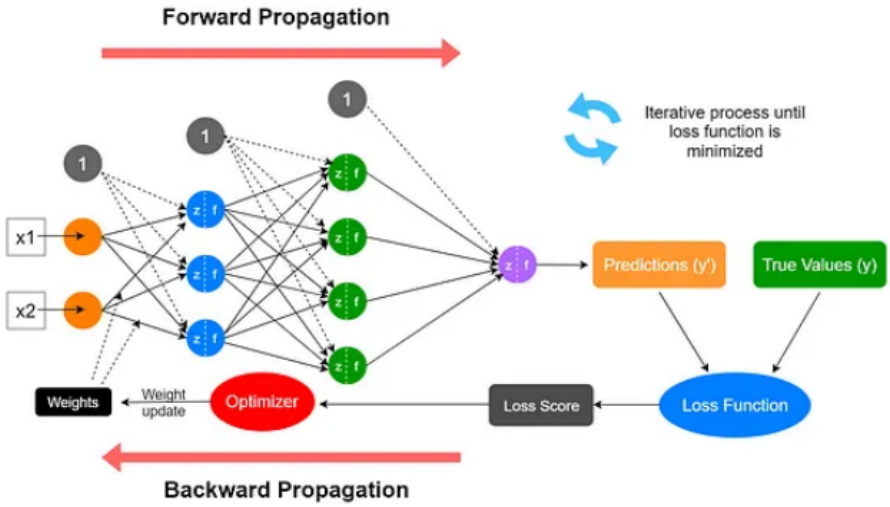


Image 5.4: *An example of an Artificial Neural Network, with a basic overview of the training process [4]*

The most commonly used activation function is the **ReLU** (Rectified Linear Unit). This nonlinear function is described by the following equation:

$$f(x) = \max(0, x) \tag{5.1}$$

This function is easy to implement, efficient and is found to successfully address the problem of vanishing gradient

A variant of **ReLU**, **Leaky ReLU** allows a small gradient when the unit is not active, and can be used to better mitigate the problem of vanishing gradients in some problems.

The **sigmoid** function has a value range between 0 and 1.

$$\sigma(x) = \frac{1}{1 + e^{-x}} \tag{5.2}$$

It is mainly used for models that make a prediction as their output, so the output of the model must produce a probability value.

The **softmax** function converts a range of input values into a probability distribution, with a sum of 1.

$$\sigma(z_i) = \frac{e^{z_i}}{\sum_{j=1}^{K} e^{z_j}} \tag{5.3}$$

It is usually used at the last layer of a classifying network, to produce the prediction values for each class.

One of the most widely used loss function is the **Mean Squared Error**. Its formula is:

$$L_{MSE} = \frac{1}{N}\sum_{i=1}^{N}(y_i - f(x_i))^2 \tag{5.4}$$

where f is the function we want to fit to the y_i target values. This metric is usually used in regression tasks, where the output of the model is some numeric value.

Mean Absolute Error or **L1 loss** is preferred to **Mean Squared Error** when there are many outliers in the training data. Its formula is:

$$L_{MAE} = \frac{1}{N}\sum_{i=1}^{N}|y_i - f(x_i)| \tag{5.5}$$

Cross Entropy Loss is the metric most commonly used in classification tasks. It measures the difference between the predicted probability distribution q and the actual distribution p of the class labels. For M number of classes, the Cross Entropy loss is calculated:

$$L_{CE} = -\sum_{i=1}^{M} p_i \log(q_i) \tag{5.6}$$

Gradient Descent is the optimization algorithm most commonly used in Neural Networks. It is usually used in training to minimize the loss function. Gradient Descent changes the parameters of the model, by readjusting them with the negative of the cost gradient.

$$\theta_{new} = \theta - \eta\nabla_\theta L(\theta) \tag{5.7}$$

with η being the learning rate.

A variation of Gradient Descent, **Stochastic Gradient Descent** takes into consideration only a single sample of the training examples when computing the gradient of the loss, and approximates for the whole training dataset. In contrast, the Gradient Descent algorithm calculates the gradients using the whole training domain. The SGD algorithm is much faster, albeit less stable in some cases due to its stochastic nature.

The **Backpropagation** algorithm consists of two steps: a forward calculation of the models predictions and a update of the layers weights, by computing the gradient of the loss backwards for each layer of the model. The partial derivatives for each parameter are calculated by iteratively applying the derivative chain rule. The process efficiently propagates the loss information to each parameter of the model, making it easier to fit to the target data.

The learning process, with the forward prediction and the backward weight updates, continues over multiple iterations or epochs, improving the predictive ability of the model, until it is stopped by a hard limit (a predetermined number of epochs) or a soft limit (a low threshold for the loss function).

The **learning rate** parameter in the gradient descent process is crucial to the outcome of the training process. A small value can delay the convergence of the model to the optimal position, or even not converge at all when the model is trained with a set amount

of epochs. On the other hand, a large value of the learning rate can cause the model to overshoot the minimum, oscillating between suboptimal positions. Thus the selection of the learning rate is linked to each problem and must be considered during the design of the training process.

A deep learning model needs to be complex enough to be able to capture the underlying patterns in the training distribution. **Underfitting** happens when the training data exceed the capabilities of the model, resulting in decreased performance. On the opposite side, **overfitting** refers to the problem of the model learning patterns from the data that do not represent structural knowledge, but noise or fragmented information. When an overfitted model is applied to new data, for example the part of data reserved for the validation or testing, performs significantly worse comparing with the training process.

We show a simple example of fitting a set of data in 5.5.

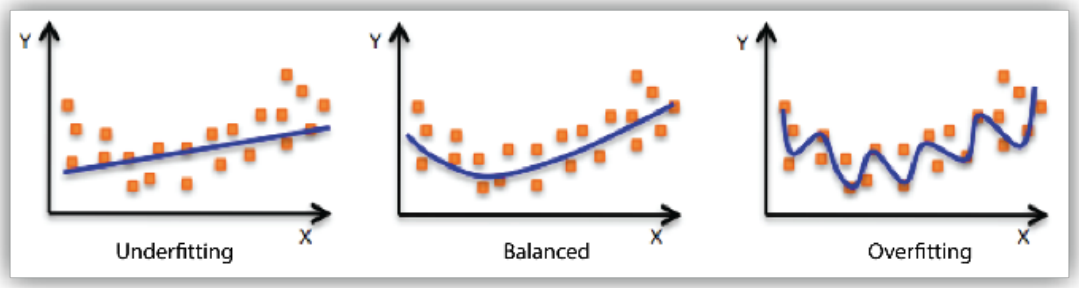


Image 5.5: *An example of underfitting and overfitting [5]*

Underfitting stems from a model of inadequate depth, and it can be easily observed when error rates are too high in both training and testing data. Thus, a solution for underfitting is usually straight-forward. Overfitting, on the other hand, is a much more difficult problem to address, as it cannot always be deduced from the performance of the model. The robustness of a model to overfitting is relevant to the generalization ability of the model.

Regularization methods aim to reduce the impact of overfitting. During training, a penalty value is added to the loss function, discouraging excessive complexity. The two most common regularization methods are L1 and L2 regularization.

In **L1** regularization (**Lasso**) the term $\lambda_1 \sum_i^N |w_i|$ is added to the loss function. It encourages sparse weight matrices, with most weights becoming zero. Neurons tend to use only a subset of their inputs, which is important to the task, and thus they are resistant to noise.

In **L2** regularization (**Ridge**) the term $\lambda_2 \sum_i^N |w_i{}^2|$ is added to the loss function. It encourages smaller weight values and the utilization of all the inputs.

Regularization methods prevent overfitting, enhancing the performance of the model in testing domains, stabilize the model against small changes and promote the robustness and safety of the model.

5.3.2 Convolutional Neural Networks

Traditional ANNs are of limited use for computer vision tasks, as they tend to struggle with the computational complexity required to process image data. They are also prone to overfitting as the number of hidden layers is increased, while most vision tasks require an increased network depth.

Convolutional Neural Networks (CNN) (5.6) are a kind of feedforward neural network that is able to extract features from data with convolution structures [18]. They are designed and used primarily for Computer Vision tasks. Their main advantages, compared to ANNs are:

- Local Connections. Each neuron is no longer connected to all neurons of the previous layer, but only to a small number of neurons, which is effective in reducing parameters and speed up convergence
- Weight sharing. A group of connections can share the same weights, which further reduces parameters.
- Down-sampling dimensionality reduction. A pooling layer harnesses the principle of image local correlation to down-sample an image, which can reduce the amount of data while retaining useful information. It can also reduce the number of parameters by essentially removing trivial features

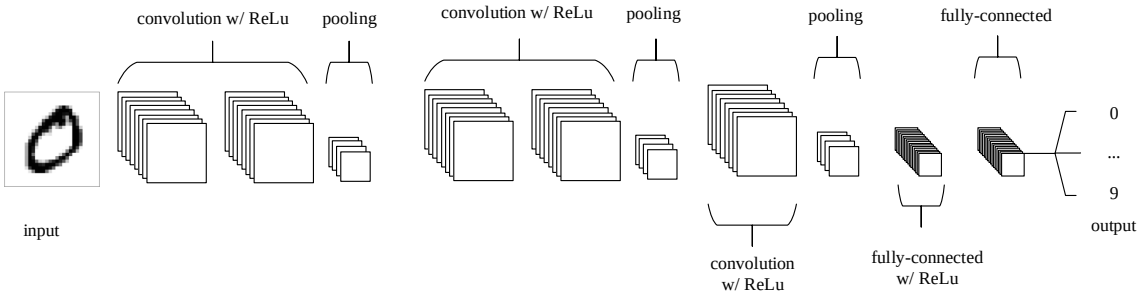


Image 5.6: *An overview of a basic CNN architecture [6]*

A typical CNN has three core types of layers [6]: the convolutional, the pooling and the fully-connected layers.

The **convolutional layer** calculates the product between the layer weights and the input, as shown in 5.7. The layer parameters correspond to convolution kernels, learnable filter structures typically with smaller spatial dimensions, that span the entirety of the input to create a activation map, highlighting certain features of the input. Comparing to ANNs, CNN neurons connect only to a small region of the input, which is referred as the receptive field size of the neuron.

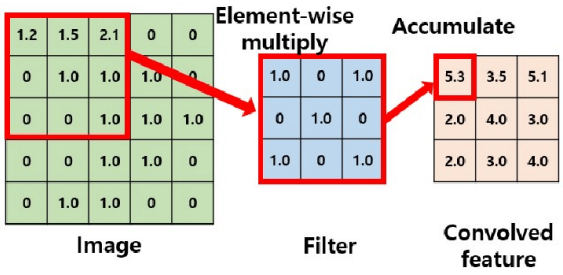


Image 5.7: *Example of convolution [7]*

Convolutional layers can be optimized using the hyperparameters depth, stride and padding. The depth is set through the number of neurons assigned to the same region of the input. Reducing depth incurs a significant reduction of the models parameters, but can also worsen the model's performance. The value of stride refers to the amount of pixels the filter is shifted after each convolution. This parameter influences the spatial dimensions of the final activation map. Padding is applied at the border of the input typically with values equal to zero and can be used to further control the output dimensions.

For each dimension of the input, the correspondent dimension of the output can be determined by the following equation:

$$W_{out} = \frac{W_{in} - K + 2P}{S} + 1 \tag{5.8}$$

where K is the correspondent kernel dimension, S is the stride in the correspondent dimension and P is the padding value.

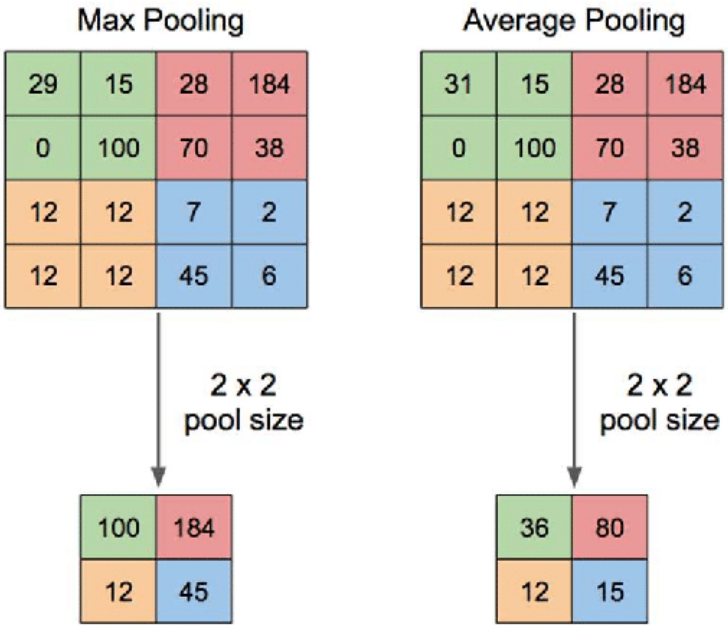


Image 5.8: *Example of pooling operations [8]*

The **pooling layer** performs downsampling along the spatial dimensions of the input, with a selection of a downsampling function such as the maximum or the average of the

region. An example of maximum and average pooling is shown in 5.8. In most CNNs max-pooling layers are utilized, with 2×2 kernels applied with stride of 2 across the input. This scales the activation map to one fourth of the original size. Alternative pooling functions can be utilized to perform L1/L2 regularization.

The **fully-connected layers** contain neurons that are densely connected to the neurons of the adjacent layers, resembling the connections of the traditional ANN. They are typically used as the output layers for classification tasks.

Except for these standard layers, other functional layers can be introduced to the training process to increase the capabilities of the model.

Batch Normalization [19] is a regularization method that can achieve higher learning rates for the training model. Batch Normalization layers are used to normalize the activation maps of each mini-batch. Given a mini-batch x, the output of the layer will be:

$$\hat{x}_i = \frac{x_i - \mu_B}{\sqrt{\sigma_B^2 + \epsilon}} \tag{5.9}$$

where the mean and variance of the minibatch can be computed by:

$$\mu_B = \frac{1}{m}\sum_{i=1}^{m} x_i \qquad \sigma_B^2 = \frac{1}{m}\sum_{i=1}^{m}(x_i - \mu_B)^2 \tag{5.10}$$

ϵ is usually initialized to a small number (on the 10^{-7}) to avoid a division with zero.

Dropout [20] is a method that select at random a specified percentage of neurons and their connections to be dropped by the network. Dropout layers significantly reduce overfitting, and are more efficient compared to other regularization methods.

5.4 Residual Convolutional Networks

The depth of convolutional neural networks seem to be crucial to the performance of the network in image classification tasks. However, depth can result in gradient vanishing or gradient exploding problems, which can have a negative impact on the learning process, with accuracy getting saturated or even degrading after convergence. This degradation problem suggests that it can be difficult for networks to approximate identity mappings from multiple nonlinear layers. These problems were usually addressed by using sufficiently low learning rates, normalized initialization of the weights or intermediate normalization layers. In order to address the degradation problem and ease the training process of deep CNNs, He et al. [21] suggested a reformulation of the convolutional layers as learning residual functions with reference to the input layers. With residual learning, if the identity mapping of a layer is optimal, it will tend to be skipped by driving its weights to zero. A block utilizing a skipping architecture is shown in 5.9.

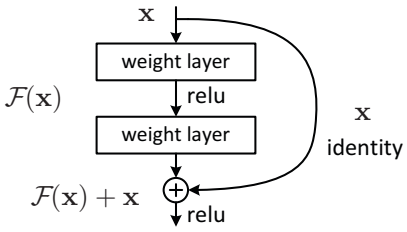


Image 5.9: *The basic building block of ResNet, showing the 'skip' architecture that resembles the residual function*

Given some stacked layers and their desired underlying mapping $\mathcal{H}(x)$ to be fitted, the stacked nonlinear layer is to fit the mapping $\mathcal{F}(x) := \mathcal{H}(x) - x$, assuming that input and output have the same dimensions. The original function becomes $\mathcal{F}(x) + x$. This can be achieved by using feedforward 'shortcut' connections, skipping a number of layers, without adding any extra parameters to the network. An outline of the ResNet-34 architecture is provided in 5.10

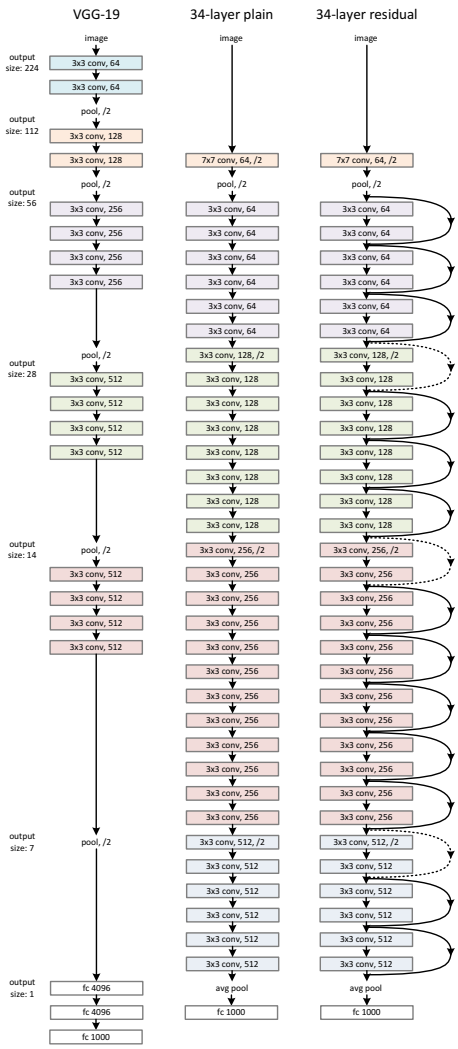


Image 5.10: *Architecture of the ResNet34 models*

The residual networks are easier to optimize, comparing with similar depth networks that simply stack layers, and they generally outperform plain networks with similar depth. Various depths of the residual networks were considered, with deeper nets showing greater performance.

5.5 Fully Convolutional Networks

Fully convolutional networks [2] expand on the semantic segmentation capabilities of CNNs. In Semantic Segmentation both local and global information impact the task performance. By extracting feature mappings from multiple layers, FCNs are able to combine deep, coarse semantic information with shallow, fine appearance information. Both learning and inference are performed with the entirety of the image by dense feedforward computation and backpropagation.

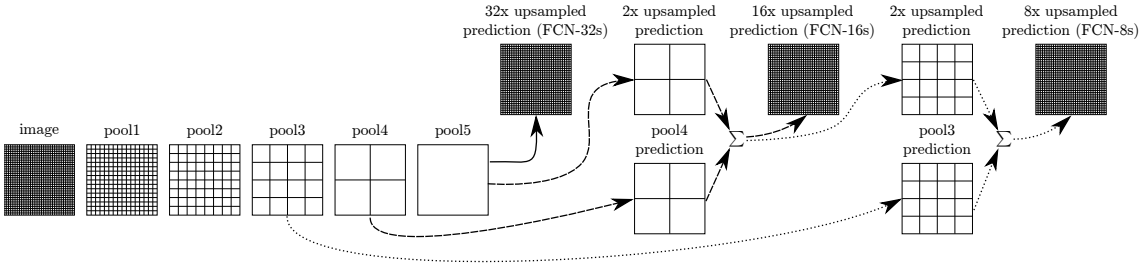


Image 5.11: *An overview of the three basic FCN architectures*

Convolutional networks are generally comprised of a convolutional, a pooling and an activation component. While a general deep net computes a general nonlinear function, a net with layers with these components computes a nonlinear filter, which is called a deep filter or fully convolutional network. A FCN naturally operates on an input of any size, and produces an output of similar (possibly resampled) spatial dimensions.

Upsampling on the filter output is done by backwards convolution, with an output stride equal to the desired factor. It is performed in network with backpropagation from pixelwise loss.

The structure of the FCN model is shown in 5.11. Typically a FCN utilizes a deep classification convolutional network as the backbone. In the paper, Long et al. adapted and tested AlexNet [37], VGG16 [38] and GoogLeNet [39]. The backbone network is integrated to the FCN pipeline without its classifier layer. At the coarse output a 1×1 convolution with channel dimension equal to the number relevant to the classification task is added, followed by a deconvolution layer to bilinearly upsample the coarse outputs to pixel-dense outputs. This pipeline performed relatively well, but the observed output was coarse, as the 32 pixel stride at the final prediction layer limited the detail of the output. This problem was addressed by the introduction of links that combine the final prediction layer with lower layers that had finer strides. Combining fine layers and coarse layers lets the model make local predictions that respect global structure. Several of these 'skips' were tried. The original FCN, with a 32 pixel stride was called FCN-32s. For the FCN-16s the original stride is halved to a 16 pixel stride layer. A 1×1 convolution layer is added to the

immediate lower pool layer, and the output is fused with the prediction made at stride 32, by adding a 2× upsampling layer and summing. The 2× upsampling is initialized to bilinear interpolation, and the parameters are allowed to be learned. With a similar process the FCN-8s is built, by fusing the 2× upsampling of the previous predictions with the prediction from the immediate lower pool layer. These non-linear architectures outperformed FCN-32s, with FCN-8s granting the best results, both quantitatively and qualitatively.

5.6 Diffusion Models

The basic ideas behind Diffusion models were introduced by Sohl-Dickstein et al. [22]. The diffusion process is defined by a forward diffusion pass, and then a reverse generative process, as is depicted in 5.12.

Ho et al. [9] provided a formal definition of Denoising Diffusion Probabilistic Models (DDPM), expanding on the framework proposed by Sohl-Dickstein et al. It is shown that Diffusion models can generate high quality samples and outperform other generative methods, such as GANs.

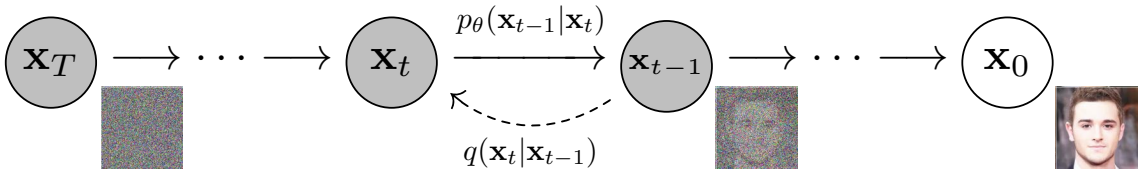


Image 5.12: *A basic overview of the Diffusion generative process [9]*

Diffusion models [9] are latent variable models of the form $p_\partial(\mathbf{x}_0) := \int p_\partial(\mathbf{x}_{0:T})\, d\mathbf{x}_{1:T}$, where $\mathbf{x}_1, \ldots, \mathbf{x}_T$ are latents of the same dimensionality as the data $\mathbf{x}_0 \sim q(\mathbf{x}_0)$.

The joint distribution $p_\partial(\mathbf{x}_{0:T})$ is called the *reverse process*, and is defined as a Markov chain with learned Gaussian transitions starting at $p(\mathbf{x}_T) = \mathcal{N}(\mathbf{x}_T; \mathbf{0}, \mathbf{I})$:

$$p_\partial(\mathbf{x}_{0:T}) := p(\mathbf{x}_T) \prod_{t=1}^{T} p_\partial(\mathbf{x}_{t-1}|\mathbf{x}_t) := p(\mathbf{x}_T) \prod_{t=1}^{T} \mathcal{N}(\mathbf{x}_{t-1}; \mu_\partial(\mathbf{x}_t, t), \Sigma_\partial(\mathbf{x}_t, t)) \tag{5.11}$$

where $p_\partial(\mathbf{x}_{0:T})$ is the Markov chain and $\mathcal{N}(\mathbf{x}; \mu, \sigma)$ is the normal distribution with mean μ and covariance σ.

What distinguishes diffusion models from other types of latent variable models is that the approximate posterior $q(\mathbf{x}_{1:T}|\mathbf{x}_0)$, called the *forward process* or *diffusion process*, is fixed to a Markov chain that gradually adds Gaussian noise to the data according to a variance schedule $\beta_1, \ldots, \beta_T$:

$$q(\mathbf{x}_{1:T}|\mathbf{x}_0) := \prod_{t=1}^{T} q(\mathbf{x}_t|\mathbf{x}_{t-1}) := \prod_{t=1}^{T} \mathcal{N}(\mathbf{x}_t;\ \sqrt{1-\beta_t}\mathbf{x}_{t-1}, \beta_t \mathbf{I}) \tag{5.12}$$

The forward process variances β_t can be learned by reparameterization or held constant as hyperparameters, and expressiveness of the reverse process is ensured in part by the choice of Gaussian conditionals in $p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)$, because both processes have the same functional form when β_t are small.

Training is performed by optimizing the variational lower bound on negative log likelihood:

$$\mathbb{E}\left[-\log p_\theta(\mathbf{x}_0)\right] \leq \mathbb{E}_q\left[-\log \frac{p_\theta(\mathbf{x}_{0:T})}{q(\mathbf{x}_{1:T}|\mathbf{x}_0)}\right] = \mathbb{E}_q\left[-\log p(\mathbf{x}_T) - \sum_{t\geq 1} \log \frac{p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)}{q(\mathbf{x}_t|\mathbf{x}_{t-1})}\right] =: L \tag{5.13}$$

Using the formulation of the forward process,:

$$q(\mathbf{x}_t|\mathbf{x}_0) = \mathcal{N}(\mathbf{x}_t;\ \sqrt{\bar{a}_t}\mathbf{x}_0, (1-\bar{a}_t)\mathbf{I}) \tag{5.14}$$

where:

$$a_t := 1 - \beta_t$$
$$\bar{a}_t := \prod_{s=1}^{t} a_s$$

L can be improved to:

$$\mathbb{E}_q\Big[\underbrace{D_{\mathrm{KL}}(q(\mathbf{x}_T|\mathbf{x}_0) \parallel p(\mathbf{x}_T))}_{L_T} + \sum_{t>1} \underbrace{D_{\mathrm{KL}}(q(\mathbf{x}_{t-1}|\mathbf{x}_t, \mathbf{x}_0) \parallel p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t))}_{L_{t-1}} \underbrace{-\log p_\theta(\mathbf{x}_0|\mathbf{x}_1)}_{L_0}\Big] \tag{5.15}$$

KL divergence is used directly to compare $p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)$ against forward process posteriors, which are tractable when conditioned on $\mathbf{x}_0$:

$$q(\mathbf{x}_{t-1}|\mathbf{x}_t, \mathbf{x}_0) = \mathcal{N}(\mathbf{x}_{t-1}; \tilde{\mu}_t(\mathbf{x}_t, \mathbf{x}_0), \tilde{\beta}_t\mathbf{I}) \tag{5.16}$$

$$\text{where} \quad \tilde{\mu}_t(\mathbf{x}_t, \mathbf{x}_0) := \frac{\sqrt{\bar{a}_{t-1}}\beta_t}{1-\bar{a}_t}\mathbf{x}_0 + \frac{\sqrt{a_t}(1-\bar{a}_{t-1})}{1-\bar{a}_t}\mathbf{x}_t \quad \text{and} \quad \tilde{\beta}_t := \frac{1-\bar{a}_{t-1}}{1-\bar{a}_t}\beta_t \tag{5.17}$$

By treating the forward process variances β_t as constant, the approximate posterior q has no learnable parameters and thus the L_T is constant. Thus, with variance fixed, L_t can be expressed as:

$$L_{t-1} = \mathbb{E}_q\left[\frac{1}{2\sigma_t^2}\|\tilde{\mu}_t(\mathbf{x}_t, \mathbf{x}_0) - \mu_\theta(\mathbf{x}_t, t)\|^2\right] + C \tag{5.18}$$

where C is a constant not dependent on θ. The training objective is formulated in minimizing the distance between the means of each distribution.

μ_∂ is parametrized to:

$$\mu_\partial(\mathbf{x}_t, t) = \tilde{\mu}_t\left(\mathbf{x}_t, \frac{1}{\sqrt{\bar{a}_t}}(\mathbf{x}_t - \sqrt{1-\bar{a}_t}\epsilon_\partial(\mathbf{x}_t))\right) = \frac{1}{\sqrt{a_t}}\left(\mathbf{x}_t - \frac{\beta_t}{\sqrt{1-\bar{a}_t}}\epsilon_\partial(\mathbf{x}_t, t)\right) \tag{5.19}$$

Then the L_{t-1} can be simplified to:

$$\mathbb{E}_{\mathbf{x}_0,\epsilon}\left[\frac{\beta_t^2}{2\sigma_t^2 a_t(1-\bar{a}_t)}\left\|\epsilon - \epsilon_\partial(\sqrt{\bar{a}_t}\mathbf{x}_0 + \sqrt{1-\bar{a}_t}\epsilon, t)\right\|^2\right] \tag{5.20}$$

So optimizing an objective resembling denoising score matching is equivalent to using variational inference to fit the finite-time marginal of a sampling chain resembling Langevin dynamics.

A simplified training objective was also proposed:

$$L_{\text{simple}}(\partial) := \mathbb{E}_{t,\mathbf{x}_0,\epsilon}\left[\left\|\epsilon - \epsilon_\partial(\sqrt{\bar{a}_t}\mathbf{x}_0 + \sqrt{1-\bar{a}_t}\epsilon, t)\right\|^2\right] \tag{5.21}$$

which is found to lead to better sample quality, as it emphasizes different reconstruction aspects that downplay terms with smaller contributions.

5.7 Contrastive Learning

Contrastive Learning [23, 24, 25] learns similar/disimilar representations from data organized in corresponding pairs. It is based on the assumption that similar instances contain information that is closer and dissimilar instances will be farther apart [26]. Contrastive Learning is able to more easily capture relevant features and similarities, suppressing dissimilar information. A visual example of Contrastive Learning is presented in 5.13.

Contrastive Learning usually leverages a data augmentation scheme, using spatial or color transformations to enhance the original dataset, that enables the model to learn the embedded information with better resilience to variations. An Encoder Network is employed to create a mapping of the enhanced dataset, which is then, with the use of a Projection Network, casted into lower-dimensional space. The Contrastive Learning Process is performed on this space, named the embedding space, with the use of an appropriate loss criterion.

Contrastive learning can be applied in Supervised and Self-Supervised settings. In Supervised Contrastive Learning labeled data is used for training the model. The objective is to create models that can minimize the distance between similar representations and maximize the distance for dissimilar features. In Self-Supervised Contrastive Learning similar/dissimilar representations are learned using unlabeled data. Typically, the unlabeled data is clustered in positive and negative pairs that can be used to help the model learn meaningful representations of the data.

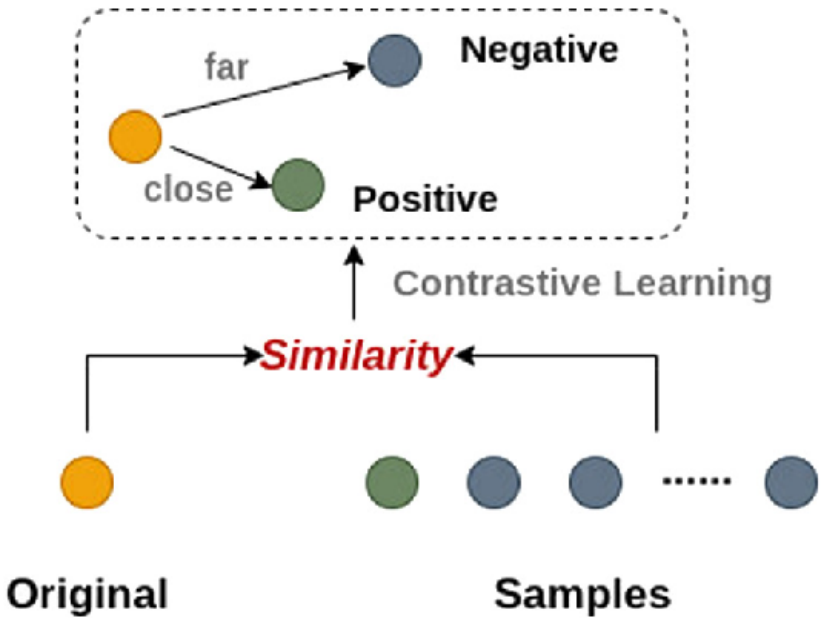


Image 5.13: *A visualization of how Contrastive Learning works [10]*

5.8 Domain Adaptation

Transfer Learning aims to help a Machine Learning model transfer the knowledge learned from one domain to another, related one.

In a typical transfer learning setting, there are two concepts: 'domain' and 'task'. A domain relates to the feature space of a specific dataset and the marginal probability distribution of features. A task relates to the label space of a dataset and an objective predictive function. The goal of transfer learning is to transfer the knowledge learned from the task T_a on domain A to the task T_b on domain B. A well-known transfer learning method is fine-tuning, where pretrained generic model are tuned-in specified tasks.

Domain Adaptation is a particular type of transfer learning. In domain adaptation, it is assumed that the domain feature spaces and tasks remain the same while the marginal distributions are different between the source and target domains [27]. Domain adaptation aims to minimize the impact of domain shift between source and target domain, which refers to the change of data distribution between the source and the target domain. [12].

Domain Adaptation methods differ based on their settings and their characteristics. To provide some categorization [27]:

- Label Availability: **Supervised & Semi-Supervised & Unsupervised.** In supervised Domain Adaptation, labeled data from the target domain are available during the training process. In semi-supervised Domain Adaptation, a small number of labeled data as well as unlabeled data in the target domain are available in the training process In unsupervised Domain Adaptation, only unlabeled target data are available for training.

- Modality Difference: **Single-Modality & Cross-Modality.** In single modality Domain Adaptation, the source and target domains share the same data modality or setting.

In cross-modality Domain Adaptation, the source and target settings are different

- Number of sources: **Single-Source & Multi-Source**. In single-source Domain Adaptation, the model is trained on a single source domain, whereas multi-source Domain Adaptation uses training samples from multiple source domains.

Unlike Domain Adaptation, *Domain Generalization* considers the case where the target domain is totally unknown, with the goal being to enhance the generalization ability of the model for any unknown target domain [12]. Domain Generalization is typically studied under two settings, **single-source** and **multi-source**. Single-source domain generalization assumes a single data distribution, i.e. a homogenous training domain. Multi-source domain generalization studies several distinct, but relevant domains.

Domain Generalization has many applications in computer vision, such as object recognition, semantic segmentation, autonomous driving and medical imaging. Many domain generalization methods have been developed that can be assigned to the following general categories [12]:

- Domain Alignment: Domain Alignment approaches aim to minimize the difference between source domains, in order to encourage the learning of domain-invariant representations. [40, 41, 42]
- Meta-Learning: Meta-learning utilizes samples from related tasks to benefit future learning. [43, 44, 45]
- Data Augmentation: Data Augmentation is used to avoid overfitting and improve generalization, by applying a transformation to the source. [46, 47, 48]
- Ensemble Learning: With Ensemble Learning multiple copies of the same model are trained, using different initialization and parameters, and their ensemble is used for prediction. [49, 50]
- Self-Supervised Learning: This method uses labels generated from the data to learn. It is mainly used with contrastive methods, to learn using minimization instance-aware representations. [51]
- Learning Disentangled Representations: This method allows the model to learn disentangled representation. [52]
- Regularization [53]
- Reinforcement Learning [54, 55, 56]

Chapter 6

Approach

In this chapter we present the relevant work in our area of interest, we describe the methods and architectures we adopted in our work and we outline our methodology. In the first part of this chapter we present influential approaches to domain adaptation and generalization, diffusion models and data augmentation. We then describe thoroughly the methods we used in our work. In the second part of the chapter we describe the structure of our model pipeline. Finally, we present the details of our implementation.

6.1 Related Work

6.1.1 Domain Adaptation & Generalization

To address the issue of domain shift [57], many domain adaptation approaches are proposed. These approaches are useful in areas where environmental conditions are highly variable (Autonomus Driving), or where the differences in capturing equipment and the difficulties in annotating images generally limit the number of available datasets (Medical Image Analysis).

The main evaluation method for domain adaptation techniques is to test the trained models on target domains semantically similar to source ones. Due to the labor-intensive nature of densely annotating real-life datasets, such as the *Cityscapes* [1] dataset, synthetic datasets of rendered urban environments, such as the *SYNTHIA* [11] or the *GTA5* [58] datasets are created to be used as the source dataset for these methods. Models trained on the synthetic datasets are evaluated with real data, as a standard measure of the model's adaptability.

A variety of learning techniques are developed, such as adversarial/contrastive learning, self-training, style augmentation and feature disentanglement.

Consistency Regularization techniques include mixing images from the target domain into source, and use these samples for training [59], leveraging extra information, such as depth, that is provided by some datasets (for example *SYNTHIA* [11]) to enhance the segmentation capabilities of standard classifiers [60]. The CAMix [61] method uses a contextual mask generation and a significance-reweighted consistency loss to identify context-dependency across domains (6.1). Tsai et al. [62] propose a patch-level alignment method, leveraging the K-means algorithm to assign each patch to clusters. Then a

K-way classifier is used to predict the cluster of each patch in the source domain. The PixMatch [63] introduces a consistency enforcing method to encourage the target's resilience to perturbations. The ROAD [64] framework utilizes the target images to imitate a pretrained real style model. It also adapts a spatial-aware method to improve the alignment between the domains.

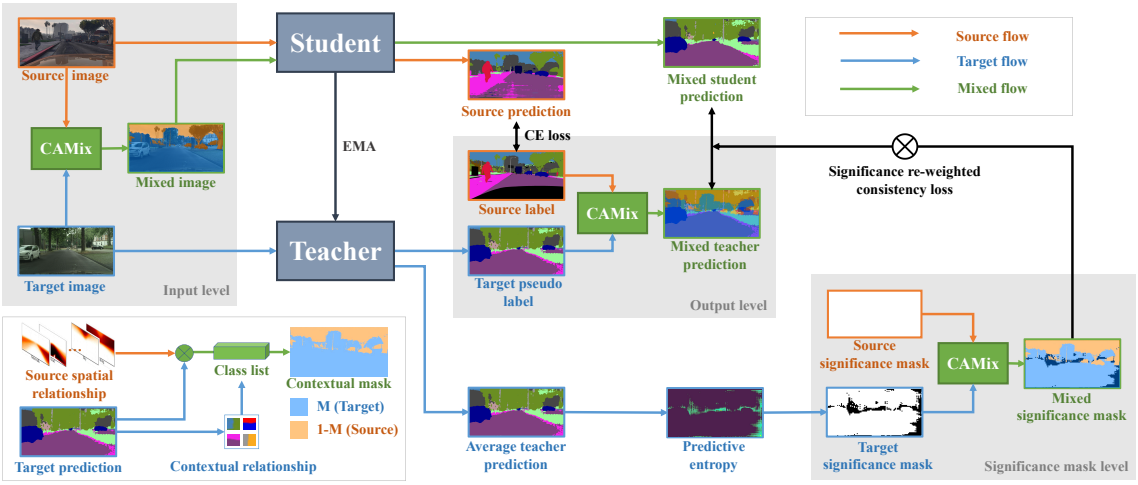


Image 6.1: *Overview of the Context Aware Mixup (CAMix) method*

Adversarial methods mainly use discriminator networks to minimize some measure of distance between source and target domain, increasing the adaptation capabilities of the model. The CyCADA [65] method combines local pixel-level and global feature structural consistency with adversarial domain adaptation (6.2). The SIM [66] method split the semantic labels into two categories, with different approaches for their alignment between source and target domains. The MSTN [67] apply pseudo labels to target samples and compare them with source samples, combining self-training with adversarial techniques. Luo et al. [68] propose a category-level adversarial network, changing the impact of the adversarial training process of each class, based on its category-level alignment between source and target domains.

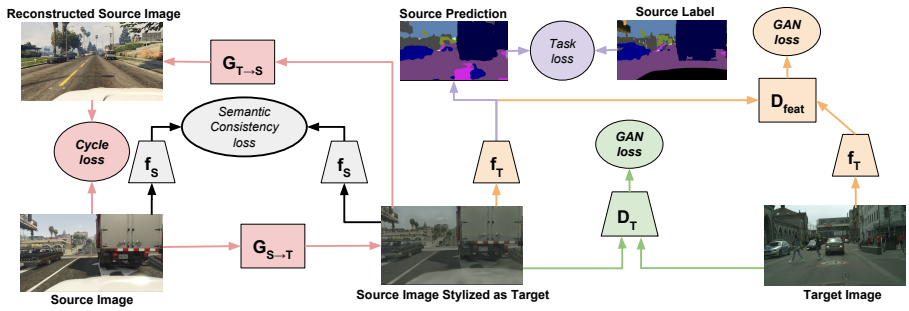


Image 6.2: *Model Architecture of the CyCADA method [12]*

Self-training methods create a pseudo-labeling of the target domain to alternatively train with the source dataset. This approach can be combined with a class-balanced [69] framework to enhance the pseudo-label generation. The DLOW [70] model is able to translate images from a source domain into intermediate domains between the source

and the target, bridging the distance between the two distributions.

Zhou et al. [71] proposes that Domain Generalization can be improved by increasing the diversity of available source domains. The Adversarial Data Augmentation [72] method utilizes a two phase iterative algorithm to achieve better generalization capabilities; a maximization phase where new images are generated and appended to the dataset and a minimization phase where the model is updated using the adversarial examples created during the maximization phase.

6.1.2 Data Augmentation

Data Augmentation methods emphasize modifications on the source data, using perturbations, replacements or merging to enhance out-of-domain robustness. Examples of data augmentation methods are presented in 6.3. PixMix [73] combines an augmentation scheme with mixing a random image from a predetermined mixing set into the original image. Cutmix [74] replaces regions of an image with a patch from another image. Other methods [75, 76] utilize the semantic structures of the images to more carefully apply image mixing. MixStyle [54] randomly selects two instances of different domains and adopts a probabilistic convex combination between instance-level feature statistics of bottom CNN layers. MaxStyle [77] uses an auxiliary image decoder attached to a segmentation network to perform self-supervised image reconstruction and style augmentation.

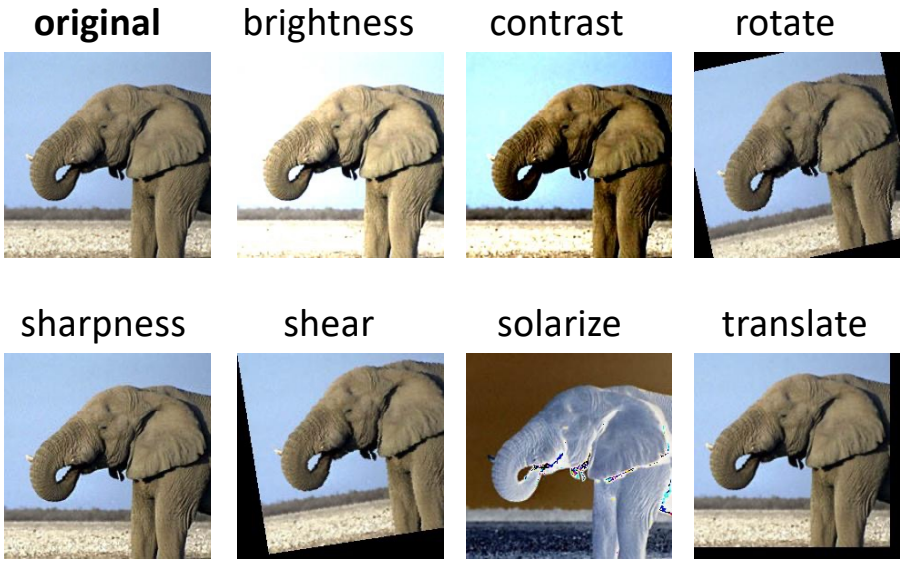


Image 6.3: *Basic image tranformations used for data augmentation [12]*

6.1.3 Diffusion Models

Diffusion models [78] seem to perform better in image generation than adversarial architectures [79] and can be used effectively in data augmentation and domain generalization tasks. Trabucco et al. [80] leverages a Stable Diffusion model to generate augmented variations of real images, by inserting and fine-tuning new tokens in the text encoder representing novel visual concepts. Data augmentation with Diffusion Models can improve

performance on classification tasks [81] and domain adaptation. DiffuMask [82] presents an automatic procedure to generate annotated synthetic images, without requiring any pixel annotations, utilizing cross-attention maps between text and image. SegDiff [13] uses Diffusion models for Image Segmentation tasks, without relying on a pre-trained backbone. U-Net encoder and decoder architectures are employed to further refine the segmented output. This method achieved promising results (6.4). In [83] is showed that Diffusion Probabilistic Models can provide meaningful representations that can be used to improve on semantic segmentation tasks. Diffusion Models are utilized in medical applications, aiming to improve anomaly detection [84], using counterfactual learning [85].

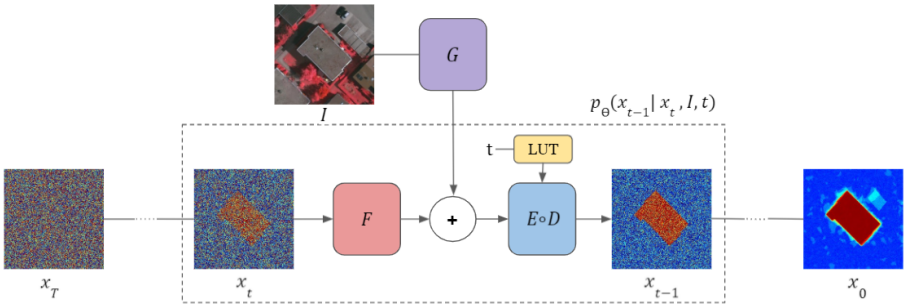


Image 6.4: *Model Architecture of the SegDiff method [13]*

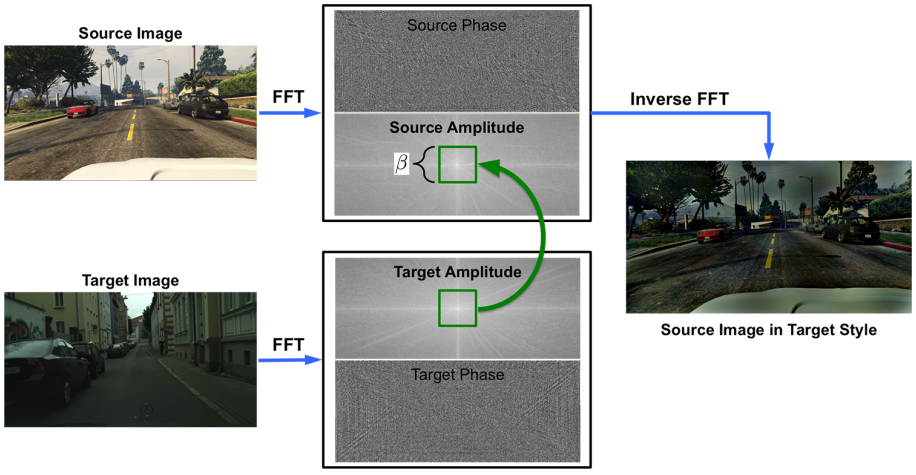


Image 6.5: *The FDA algorithm, showing the result of the frequencies substitution [14]*

6.1.4 Fourier Domain Adaptation (FDA)

The FDA [14] method can be classified as semi-supervised domain adaptation, based on entropy minimization. Its motivation is based on the existence of the majority of the photometric information of an image in low–frequency domains. By replacing this band of frequencies with the counterpart of the target, information about the outer domain can be shared with the source. The method utilizes the FFT algorithm to compute the Fourier Transform of an input image, then replaces the low-amplitude frequencies of the source image with the corresponding target image frequencies and finally reconstructs the augmented source image with the inverse Fourier Transform. A parameter β is used as

a threshold to select the frequencies to be replaced. An overview of the method is shown in 6.5. The training is conducted with these reconsituted images, and their original annotations.

6.1.5 The Saliency-balancing Location-scale Augmentation (SLAug)

Previous approaches in improving Source Domain Generalization applied global transformations or random augmentation on the source images. These tend to limit the diversity of the augmented images and may impact the generalization performance. The Saliency–balancing Location–scale Augmentation (SLAug) [28] method combines a both global and local augmentation scheme with gradient distribution harmonizing mechanism. The structure of the algorithm is shown in 6.6.

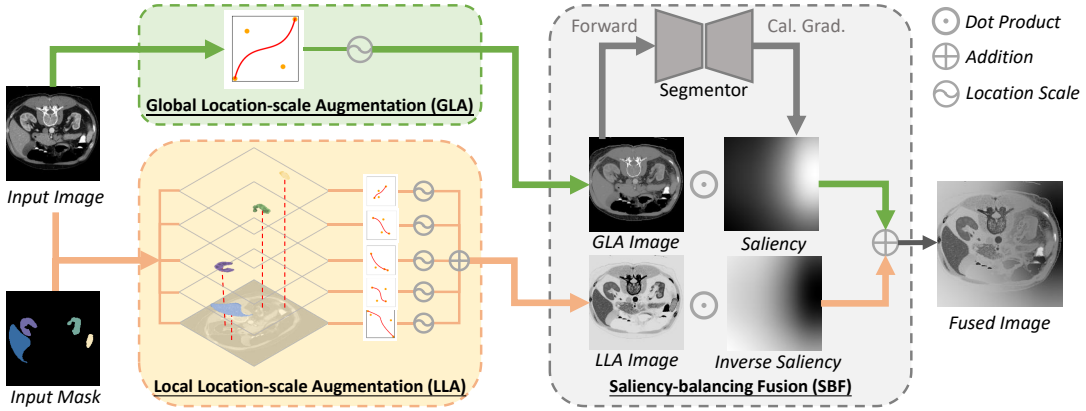


Image 6.6: *Architecture of the SLAUG pipeline*

The Location–scale Augmentation is done in two parts.

The Global Location–scale Augmentation (**GLA**) increases the source–like images through global distribution shifting. The Cubic Bézier Curve [29], a smooth and monotonic function, is utilized for the non-linear transformation of the images. The function is constrained by the highest (v_{high}) and lowest (v_{low}) value of the image region:

$$Bézier(t) = \sum_{k=0}^{3} (1-t)^{3-k} t^{kP_k}, t \in [v_{\text{low}}, v_{\text{high}}], \tag{6.1}$$

where t is a fractional value, $P_0 = (v_{\text{low}}, v_{\text{low}})$ and $P_3 = (v_{\text{high}}, v_{\text{high}})$ are the start and end points to limit the range of values, and P_1, P_2 are the control points whose values are randomly generated from $[v_{\text{low}}, v_{\text{high}}]$. Before augmentation, x is min-max normalized in the range of [0, 1]. GLA performing global distribution shifting can be described as:

$$\text{GLA}(x) = a\mathcal{F}_0(x) + \beta, \tag{6.2}$$

where $a \sim \mathcal{TN}(1, \sigma_1)$, $\beta \sim \mathcal{TN}(0, \sigma_2)$ are location-scale factors, and σ_1 , σ_2 are standard deviations of the two truncated Gaussian distributions respectively. By using $\mathcal{F}_0$, the probability of inverse is zero to ensure the GLA samples have similar appearance with original images.

The Local Location−scale Augmentation (**LLA**) takes the individual class-level regions as the processing unit and applies transformation respectively. The augmented regions are then combined linearly. This process can be represented as:

$$\mathrm{LLA}(x, m) = \sum_{c=1}^{C} a_c \mathcal{F}_{\mathrm{p}_c}(x^c) + \beta_c, \tag{6.3}$$

where $a_c \sim \mathcal{TN}(1, \sigma_1)$ and $\beta_c \sim \mathcal{TN}(0, \sigma_2)$ are location-scale factors. For all classes, $\mathrm{p}_c = 0.5$ is set to apply random inversion, except $\mathrm{p}_1 = 1$ to ensure the LLA augmented images are dissimilar to the GLA samples.

The Saliency-balancing Fusion aims to preserve the large-gradient areas that indicate to informational sensitivity. The saliency map is obtained by taking the l_2 norm of gradient values across input channels and then downsampled to the grid size of $g \times g$ followed by interpolation to the original image size via quadratic B-spline kernels for smoothing.

The following algorithm describes the proposed pipeline:

Require: Training data (x, m), network f_∂, loss function $\mathcal{L}$, global location-scale augmentation GLA, local location-scale augmentation LLA, common augmentation F.

1: $x^g \leftarrow \mathrm{GLA}(x)$
2: $x^l \leftarrow \mathrm{LLA}(x, m)$
3: $\tilde{x}^g, \tilde{x}^l, \tilde{m} \leftarrow F(x^g, x^l, m)$ ▷ Common augmentation
4: Calculate gradient $Grad = \nabla_{\tilde{x}^g} \mathcal{L}(f_\partial(\tilde{x}^g), \tilde{m})$
5: $\mathbf{s} \leftarrow normalize(smooth(|Grad|))$ ▷ Calculate saliency
6: $\tilde{x}^{fused} \leftarrow \mathbf{s} \odot \tilde{x}^g + (1 - \mathbf{s}) \odot \tilde{x}^l$
7: **return** $\tilde{x}^{fused}$

6.1.6 Channel-Level Contrastive Single Domain Generalization (CCSDG)

The main problem of Single Domain Generalization is the difficulty of generalizing in target−domains, as the segmentation models tend to overfit the source−domain. The CCSDG [30] method utilizes a contrastive approach to achieve better performance comparing to other methods. Using the shallower features of the original images and their style−augmented counterparts, the Channel Feature Disentanglement module learns to distinguish between structure relevant features, that are shared across domains, and structure−irrelevant or style representational channels, that are more sensitive to each domain characteristics and do not contribute to the models generalization abilities.

This method utilizes:

- A segmentation backbone
- A Style Augmentation (StyleAug) module that generates style−differentiations of the source image
- A Contrastive Feature Disentanglement module that performs contrastive training on the features of the source image, extracted from a convolutional layer and its style−augmented counterpart

The overview of the CCSDG pipeline is presented in figure 6.7.

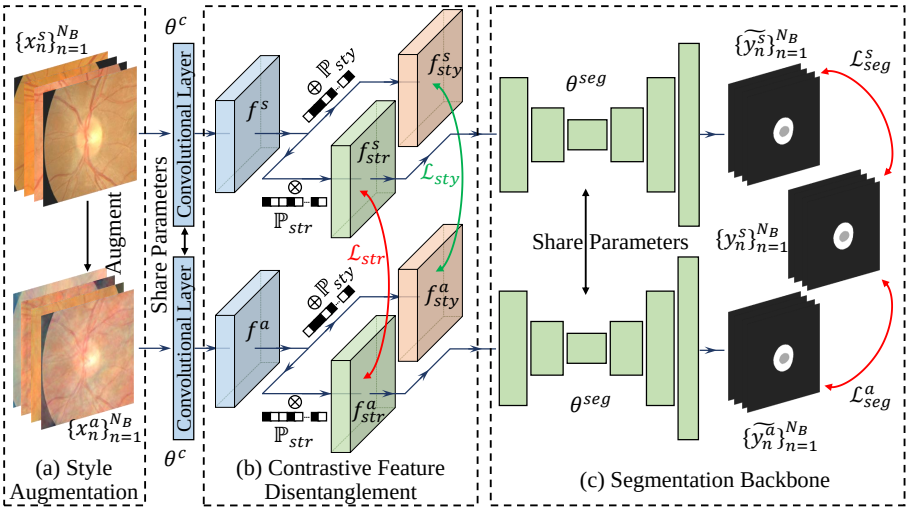
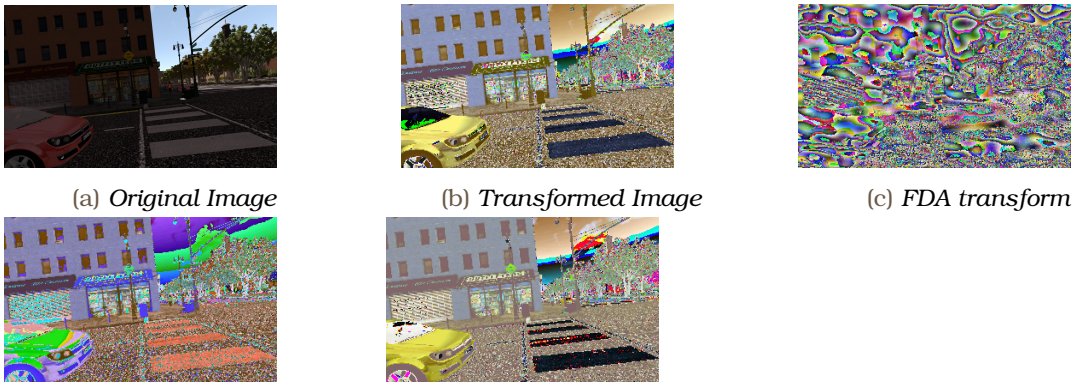


Image 6.7: *Architecture of the CCSDG pipeline*



(a) *Original Image* (b) *Transformed Image* (c) *FDA transform*

(d) *LLA transform* (e) *GLA transform*

Image 6.8: *Example outputs of the Style Augmentation Module, using an image from the SYNTHIA [11] dataset.*

The Style Augmentation module utilizes approaches such as gamma correction and noise additions from BigAug [31], the Bezier curve transformations from SLAug [28] and low–frequency compontents replacement [14] to create three corresponding augmented batches from each source batch, perform contrastive training and segmentation. The output example of the Style Augmentation module is presented in figure 6.8

The Contrastive Feature Disentanglement module uses a convolutional layer to extract from the source and the augmented images 64–channel feature maps. Using a channel mask prompt $\mathbb{P} \in \mathbb{R}^{2x64}$, that is structured to resemble a binary–element vector, each feature map is seperated into a style representation f_{sty} and a structure representation f_{str}. For the contrastive training, the style representations between original and augmented images are expected to be dissimilar and the structure representations to be similar. Two loss functions are designed:

$$\begin{cases} \mathcal{L}_{str} = \sum |Proj(f^s_{str}) - Proj(f^a_{str})| \\ \mathcal{L}_{sty} = -\sum |Proj(f^s_{sty}) - Proj(f^a_{sty})|, \end{cases} \tag{6.4}$$

where ^{s} notates the source feature maps and ^{a} the augmented ones. *Proj* with parameters ∂^{p} reduces the dimension of the feature maps.

The structural representations are finally fed to the segmentation backbone ∂^{seg} to produce the predictions. In this way, during training $\mathcal{L}_{seg}$ is minimized to optimize $\{\partial^{c}, \mathbb{P}, \partial^{seg}\}$, and $\mathcal{L}_{str} + \mathcal{L}_{sty}$ is minimized to optimize $\{\mathbb{P}, \partial^{p}\}$.

6.1.7 DiffuseMix

DiffuseMix is a data augmentation method that utilizes a diffusion model to enhance an original image dataset. The image augmentation follows three steps: generation, concatenation and fractal blending. A visualization of the algorithm is shown in 6.9. At the first step a pretrained diffusion model is guided by a selection of conditional prompts to generate diverse samples from the original images. Then, the generated images are combined with the original ones, using a set of masks that combine a part of one image with the remaining part of the other, so that the concatenated image is a hybrid between the two. The final step is to infuse in the concatenated product a random image from a dataset of self–similarity fractals.

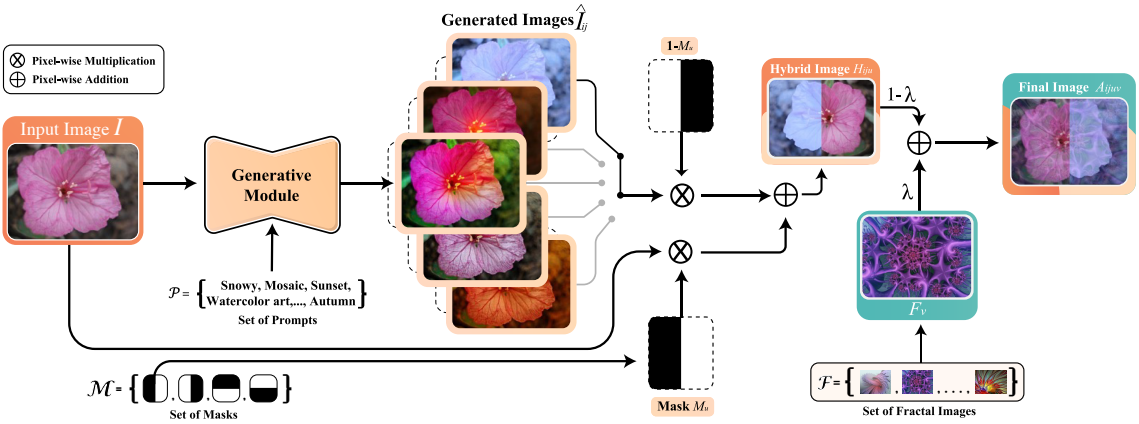


Image 6.9: *The DiffuseMix Architecture*

This approach to image-mixing provides a method that retains the basic structure of the image and provides the context needed for better augmentation. For the generation step the diffusion model InstructPix2Pix [32] is used. The prompts used were selected for their generic nature and low interference with the basic structure of a wide variation of images (figure 6.10).

The DiffuseMix architecture is capable of improving the generalization capability of ResNet-50, returning consistent performance gains across many datasets.

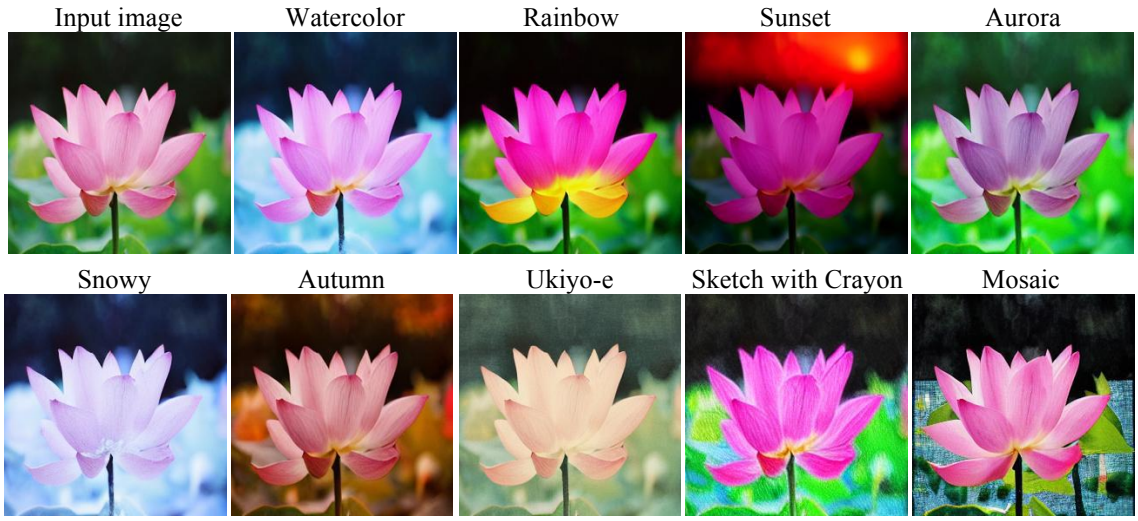


Image 6.10: *Example of the output of the diffusion model, with the corresponding textual prompts*

6.2 Methodology

6.2.1 Source Dataset

For the training we used the SYNTHIA [11], a collection of synthetic annotated images of virtual urban environments. Each image is pixel-level annotated to 13 classes. We created the training and testing datasets using the SYNTHIA-SEQS datasets, which are comprised of caption sequences of four video simulations acquired from the viewpoint of a virtual car. We selected the OMNI–L viewpoint for our experiments. From the 5 available environments (Highway, New York City, Old European Town, New York City (2), Highway (2)) available, we created 3 training datasets from the first environment, sampling 800 random images corresponding to the conditions (DAWN, FOG, SPRING), and 20 testing datasets from the rest, sampling 240 images from the conditions (DAWN, FOG, NIGHT, SPRING, WINTER). The data and label images were resized from the original size of 1280×760 to a size of 320×192, using the **cv2.resize** function provided by the **OpenCV** python library, with the interpolation parameter set to the value **cv2.INTER_NEAREST**.

6.2.2 Training Dataset

We used the pretrained DiffuseMix pipeline [33] to create enhanced variations of the original datasets. For each of the 3 conditions we applied a subset of the model's designated textual prompts. From the outputs of this pipeline, we utilize the **generated** product of the InstructPix2Pix [32] Diffusion model and the **blended** image that are the product of combining the generated image with the original, and then blending it with a random fractal image. Using an image from the SYNTHIA dataset, we present the outputs in figures 6.11, 6.12, 6.13, 6.14.

Image 6.11: *Original, Generated and Blended image outputs of the DiffuseMix pipeline, providing the text prompt 'autumn'*

Image 6.12: *Original, Generated and Blended image outputs of the DiffuseMix pipeline, providing the text prompt 'snowy'*

Image 6.13: *Original, Generated and Blended image outputs of the DiffuseMix pipeline, providing the text prompt 'sunset'*

Image 6.14: *Original, Generated and Blended image outputs of the DiffuseMix pipeline, providing the text prompt 'ukiyo-e'*

6.2.3 Training Model

The model used is a modified version of the pretrained **fcn_resnet50** that is provided in the **torchvision.models** library. This model consists of a resnet50 [21] backbone, with a FCN−32 [2] classifier. We adjusted the model to accommodate the Channel Feature Disentanglement Module [30] after the first convolutional layer of the backbone.

We also use a Projector module, as a part of the contrastive training, using the architecture described in [30].

6.2.4 Training Parameters

The training process has a duration of 120 epochs, with a batch size equal to 8. The Adam optimizer was selected for both the model and the projector module optimizers, with the learning rate set to 10^{-4} for both, and a weight decay value of 10^{-5} for the model optimizer.

The **l1_loss** [34] criterion, with the reduction parameter set to 'mean', is used for training the Channel Feature Disentanglement module, as in [30]. This function computes the mean absolute error (MAE) between each element.

We selected the function **CrossEntropyLoss** [35] as our loss criterion for the model training, with the default parameters. The loss value for each step is calculated by the following equation:

$$l(x, y) = \sum_{n=1}^{N} \frac{l_n}{N} \qquad l_n = -\log \frac{\exp(x_{n,y_n})}{\sum_{c=1}^{C} \exp(x_{n,y_n})} \tag{6.5}$$

where C equals to the number of classes and N is the size of the minibatch.

For the evaluation, we used the Mean Intersection Over Union (MIOU) metric, which is defined as the average value of the intersection between the predicted and the ground truth label annotation for each semantic class, divided by their union.

$$J(A, B) = \frac{|A \cap B|}{|A \cup B|} \tag{6.6}$$

For the metrics calculation we are using the **MulticlassJaccardIndex** [36] function from the **torchmetrics** library with default parametres apart from ignoring the background label.

6.2.5 Training

We train the model, following the steps outlined in the CCSDG [30] pipeline. In the Style Augment module we omit the gamma correction and noise transforms that the original pipeline applies to the input images.

Each epoch had a duration of an average of 40 seconds for the simple model training and an average of 160 seconds for the advanced model.

The experiments were conducted in a cloud computing environment provided by Amazon Web Services, utilizing a NVIDIA A10G GPU.

Chapter 7

Experiment

In this chapter we explore the performance of various configurations, based on the pipeline architectures we outlined in the previous chapter. We show the most interesting results of applying these data transforms and pipeline alterations. Each model was trained multiple times, with the final performance being the average of all runs.

7.1 Results from training the base model

At first we train the **fcn_resnet50** model, without any of the mentioned modifications, on the datasets created by the DiffuseMix pipeline.

We notate the baseline as the performance of the model trained on the original dataset. On the 'mode' column we notate '**gen**' as the **Generated** output and '**bl**' as the **Blended** output of the DiffuseMix pipeline, with the corresponding textual prompt. The rest of the columns represent the testing results for each condition on a domain.

An example of training loss and accuracy is provided in figure 7.1. The performance of each modality is presented in the tables 7.1, 7.2, 7.3, 7.4.

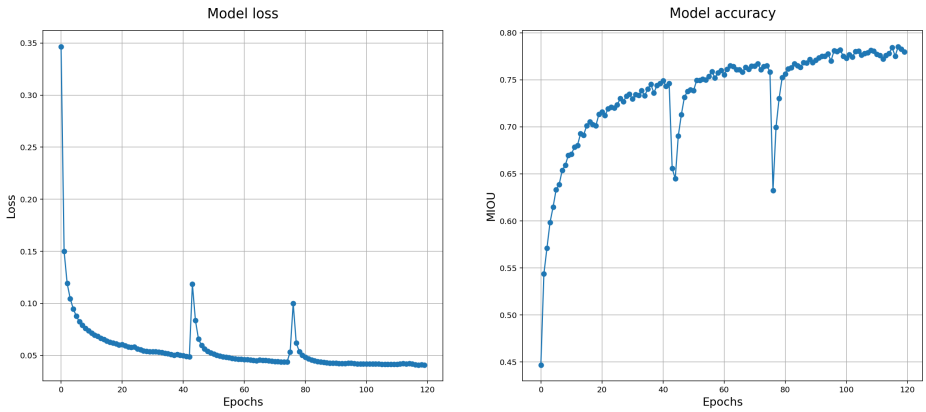


Image 7.1: *Diagrams of Loss and MIOU accuracy in training*

New York City						
Train Domain	Mode	DAWN	FOG	NIGHT	SPRING	WINTER
DAWN	baseline	**0.305**	**0.282**	**0.249**	**0.288**	**0.268**
	gen–autumn	0.264	0.241	0.224	0.241	0.231
	gen–sunset	0.255	0.246	0.202	0.241	0.221
	gen–ukiyo-e	0.256	0.232	0.209	0.245	0.228
	bl–autumn	0.243	0.215	0.184	0.218	0.193
	bl–sunset	0.239	0.212	0.186	0.209	0.180
	bl–ukiyo-e	0.247	0.231	0.189	0.235	0.208
FOG	baseline	**0.255**	**0.293**	**0.190**	0.252	**0.265**
	gen–autumn	0.219	0.236	0.155	0.217	0.213
	gen–sunset	0.171	0.223	0.126	0.178	0.178
	gen–ukiyo-e	0.232	0.250	0.173	**0.257**	0.228
	bl–autumn	0.194	0.218	0.113	0.197	0.179
	bl–sunset	0.187	0.220	0.099	0.198	0.184
	bl–ukiyo-e	0.197	0.225	0.128	0.219	0.193
SPRING	baseline	**0.246**	**0.254**	**0.200**	**0.265**	0.252
	gen–autumn	0.221	0.219	0.196	0.242	0.248
	gen–sunset	0.221	0.226	0.165	0.226	0.242
	bl–autumn	0.218	0.232	0.182	0.242	**0.256**
	bl–sunset	0.206	0.235	0.177	0.224	0.240
	bl–ukiyo-e	0.210	0.236	0.181	0.238	0.247

Table 7.1: *Training the base model on New York City*

Old European Town						
Train Domain	Prompt	DAWN	FOG	NIGHT	SPRING	WINTER
DAWN	baseline	**0.281**	**0.329**	**0.304**	**0.346**	**0.269**
	gen–autumn	0.226	0.264	0.247	0.266	0.228
	gen–sunset	0.240	0.270	0.232	0.266	0.216
	gen–ukiyo-e	0.221	0.263	0.238	0.265	0.226
	bl–autumn	0.186	0.209	0.192	0.221	0.175
	bl–sunset	0.193	0.218	0.200	0.215	0.166
	bl–ukiyo-e	0.210	0.248	0.229	0.250	0.205
FOG	baseline	**0.244**	**0.331**	**0.224**	**0.290**	**0.264**
	gen–autumn	0.224	0.272	0.201	0.262	0.219
	gen–sunset	0.197	0.231	0.162	0.181	0.173
	gen–ukiyo-e	0.217	0.281	0.218	0.282	0.222
	bl–autumn	0.193	0.227	0.165	0.211	0.170
	bl–sunset	0.206	0.229	0.157	0.212	0.174
	bl–ukiyo-e	0.196	0.231	0.193	0.237	0.183
SPRING	baseline	**0.286**	**0.289**	**0.263**	**0.309**	**0.266**
	gen–autumn	0.244	0.247	0.230	0.258	0.242
	gen–sunset	0.266	0.258	0.229	0.262	0.265
	bl–autumn	0.225	0.216	0.208	0.241	0.208
	bl–sunset	0.209	0.216	0.191	0.237	0.204
	bl–ukiyo-e	0.227	0.223	0.223	0.251	0.221

Table 7.2: *Training the base model on Old European Town*

New York City (2)						
Train Domain	Prompt	DAWN	FOG	NIGHT	SPRING	WINTER
DAWN	baseline	**0.374**	**0.303**	**0.318**	**0.386**	**0.282**
	gen–autumn	0.328	0.277	0.263	0.304	0.238
	gen–sunset	0.323	0.251	0.216	0.295	0.197
	gen–ukiyo-e	0.337	0.263	0.264	0.324	0.231
	bl–autumn	0.297	0.240	0.191	0.274	0.200
	bl–sunset	0.303	0.248	0.215	0.269	0.183
	bl–ukiyo-e	0.312	0.267	0.231	0.287	0.207
FOG	baseline	**0.316**	**0.358**	**0.257**	0.358	**0.289**
	gen–autumn	0.264	0.308	0.220	0.335	0.255
	gen–sunset	0.252	0.297	0.201	0.303	0.211
	gen–ukiyo-e	0.296	0.320	0.244	**0.359**	0.256
	bl–autumn	0.254	0.299	0.197	0.316	0.231
	bl–sunset	0.252	0.292	0.197	0.300	0.222
	bl–ukiyo-e	0.244	0.290	0.211	0.307	0.229
SPRING	baseline	**0.325**	**0.314**	**0.311**	**0.381**	0.299
	gen–autumn	0.305	0.301	0.295	0.349	**0.303**
	gen–sunset	0.309	0.268	0.250	0.371	0.284
	bl–autumn	0.285	0.293	0.276	0.347	0.287
	bl–sunset	0.281	0.292	0.269	0.314	0.264
	bl–ukiyo-e	0.289	0.300	0.290	0.325	0.281

Table 7.3: *Training the base model on New York City (2)*

Highway (2)						
Train Domain	Prompt	DAWN	FOG	NIGHT	SPRING	WINTER
DAWN	baseline	**0.415**	0.347	**0.356**	**0.389**	**0.386**
	gen–autumn	0.398	**0.388**	0.355	0.355	0.369
	gen–sunset	0.402	0.309	0.283	0.341	0.317
	gen–ukiyo-e	0.402	0.330	0.337	0.344	0.359
	bl–autumn	0.384	0.365	0.277	0.347	0.339
	bl–sunset	0.388	0.357	0.285	0.350	0.319
	bl–ukiyo-e	0.396	0.384	0.329	0.346	0.347
FOG	baseline	0.332	**0.424**	0.305	0.367	**0.388**
	gen–autumn	0.290	0.389	**0.389**	0.337	0.358
	gen–sunset	0.293	0.372	0.260	0.289	0.332
	gen–ukiyo-e	**0.363**	0.398	0.287	**0.387**	0.370
	bl–autumn	0.330	0.367	0.261	0.382	0.337
	bl–sunset	0.327	0.365	0.272	0.373	0.316
	bl–ukiyo-e	0.322	0.365	0.291	**0.387**	0.333
SPRING	baseline	**0.385**	**0.374**	**0.372**	**0.451**	**0.392**
	gen–autumn	0.369	0.357	0.368	0.429	0.365
	gen–sunset	0.383	0.319	0.313	0.423	0.358
	bl–autumn	0.324	0.325	0.347	0.417	0.343
	bl–sunset	0.352	0.357	0.314	0.424	0.344
	bl–ukiyo-e	0.322	0.365	0.291	0.387	0.333

Table 7.4: *Training the base model on Highway (2)*

We observe that the **Generated** datasets perform better than their **Blended** counterparts. Apart from some cases, the model trained on the original dataset performs

distinctly better. There seems to be a performance improvement for some conditions or combinations, especially in the **FOG** train condition on the **HIGHWAY (2)** testing environment, but we cannot safely deduce a pattern.

7.2 Results from training the augmented model

Using the same naming scheme for the column 'Mode', we show the performance of the model including the Contrastive Learning module, testing various textual prompts. As before, we compare the augmented datasets with the original (noted as 'augment'), plus the performance of the base model trained on the original data (noted as 'baseline').

An example of training loss and accuracy is provided in figure 7.2. The performance of each modality is presented in the tables 7.5, 7.6, 7.7, 7.8.

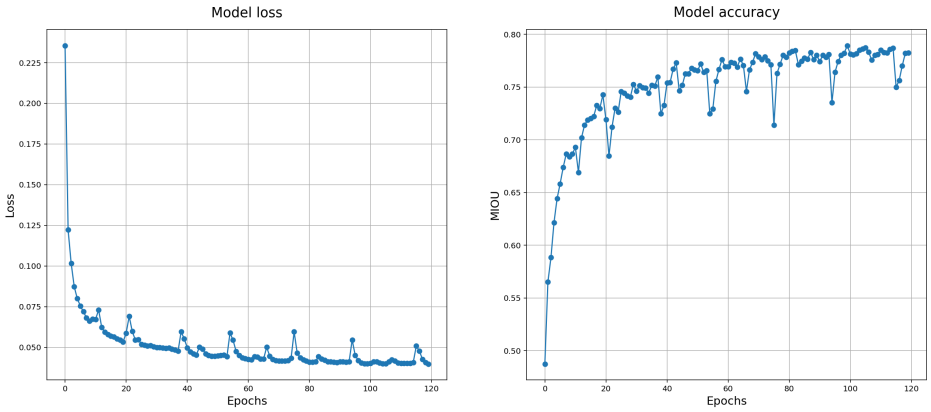


Image 7.2: *Diagrams of Loss and MIOU accuracy in training*

New York City						
Train Domain	Mode	DAWN	FOG	NIGHT	SPRING	WINTER
DAWN	baseline	**0.305**	**0.282**	**0.249**	**0.288**	**0.268**
	augment	0.228	0.220	0.176	0.196	0.185
	gen–autumn	0.190	0.182	0.158	0.169	0.174
	gen–snowy	0.190	0.203	0.160	0.202	0.195
	gen–sunset	0.228	0.220	0.179	0.208	0.189
	bl–autumn	0.216	0.191	0.174	0.186	0.167
	bl–snowy	0.222	0.206	0.182	0.210	0.192
	bl–sunset	0.216	0.208	0.168	0.193	0.178
FOG	baseline	**0.255**	**0.293**	**0.190**	**0.252**	**0.265**
	augment	0.186	0.224	0.131	0.188	0.176
	gen–autumn	0.210	0.239	0.158	0.224	0.194
	gen–snowy	0.148	0.191	0.103	0.171	0.168
	gen–aurora	0.129	0.191	0.085	0.167	0.167
	bl–autumn	0.164	0.214	0.107	0.193	0.164
	bl–snowy	0.142	0.197	0.099	0.183	0.161
	bl–aurora	0.158	0.218	0.097	0.205	0.187
SPRING	baseline	**0.246**	**0.254**	**0.200**	**0.265**	0.252
	augment	0.227	0.249	0.190	0.262	**0.255**
	gen–autumn	0.211	0.227	0.186	0.226	0.242
	gen–snowy	0.152	0.194	0.143	0.172	0.200
	gen–aurora	0.162	0.205	0.141	0.202	0.227
	bl–autumn	0.180	0.229	0.132	0.230	0.239
	bl–snowy	0.135	0.193	0.111	0.193	0.210
	bl–aurora	0.170	0.211	0.145	0.213	0.229

Table 7.5: *Training the augmented model on New York City*

Old European Town						
Train Domain	Mode	DAWN	FOG	NIGHT	SPRING	WINTER
DAWN	baseline	**0.281**	**0.329**	**0.304**	**0.346**	**0.269**
	augment	0.237	0.269	0.234	0.246	0.205
	gen–autumn	0.162	0.192	0.171	0.189	0.178
	gen–snowy	0.193	0.212	0.187	0.211	0.202
	gen–sunset	0.195	0.223	0.201	0.223	0.205
	bl–autumn	0.166	0.205	0.182	0.199	0.177
	bl–snowy	0.180	0.206	0.188	0.214	0.195
	bl–sunset	0.168	0.196	0.175	0.193	0.172
FOG	baseline	**0.244**	**0.331**	**0.224**	**0.290**	**0.264**
	augment	0.227	0.242	0.190	0.230	0.193
	gen–autumn	0.224	0.231	0.188	0.213	0.193
	gen–snowy	0.182	0.179	0.148	0.184	0.190
	gen–aurora	0.191	0.194	0.164	0.177	0.163
	bl–autumn	0.206	0.201	0.161	0.181	0.144
	bl–snowy	0.188	0.187	0.147	0.185	0.173
	bl–aurora	0.197	0.185	0.152	0.178	0.160
SPRING	baseline	**0.286**	**0.289**	**0.263**	**0.309**	0.266
	augment	0.262	0.257	0.226	0.267	**0.272**
	gen–autumn	0.242	0.238	0.219	0.241	0.249
	gen–snowy	0.179	0.170	0.148	0.182	0.200
	gen–aurora	0.208	0.195	0.177	0.198	0.208
	bl–autumn	0.223	0.191	0.169	0.203	0.212
	bl–snowy	0.181	0.148	0.130	0.175	0.187
	bl–aurora	0.191	0.172	0.154	0.182	0.188

Table 7.6: *Training the augmented model on Old European Town*

New York City (2)						
Train Domain	Mode	DAWN	FOG	NIGHT	SPRING	WINTER
DAWN	baseline	**0.374**	**0.303**	**0.318**	**0.386**	**0.282**
	augment	0.318	0.272	0.242	0.300	0.198
	gen–autumn	0.270	0.228	0.210	0.262	0.187
	gen–snowy	0.282	0.252	0.246	0.282	0.213
	gen–sunset	0.297	0.253	0.214	0.278	0.199
	bl–autumn	0.287	0.234	0.215	0.249	0.188
	bl–snowy	0.282	0.242	0.223	0.262	0.193
	bl–sunset	0.271	0.229	0.197	0.251	0.183
FOG	baseline	**0.316**	**0.358**	0.257	**0.358**	**0.289**
	augment	0.282	0.312	**0.278**	0.355	0.229
	gen–autumn	0.282	0.299	0.254	0.325	0.239
	gen–snowy	0.232	0.273	0.247	0.302	0.230
	gen–aurora	0.204	0.253	0.211	0.255	0.210
	bl–autumn	0.230	0.279	0.214	0.301	0.218
	bl–snowy	0.229	0.290	0.246	0.298	0.227
	bl–aurora	0.230	0.287	0.242	0.315	0.233
SPRING	baseline	**0.325**	0.314	**0.311**	0.381	0.299
	augment	0.304	**0.330**	0.294	**0.385**	0.313
	gen–autumn	0.306	0.314	0.300	0.349	**0.330**
	gen–snowy	0.250	0.274	0.276	0.302	0.269
	gen–aurora	0.257	0.279	0.268	0.324	0.275
	bl–autumn	0.254	0.284	0.266	0.340	0.307
	bl–snowy	0.217	0.266	0.257	0.295	0.271
	bl–aurora	0.245	0.277	0.258	0.304	0.273

Table 7.7: *Training the augmented model on New York City (2)*

Highway (2)						
Train Domain	Mode	DAWN	FOG	NIGHT	SPRING	WINTER
DAWN	baseline	**0.415**	0.347	**0.356**	**0.389**	**0.386**
	augment	0.408	**0.379**	0.323	0.354	0.346
	gen–autumn	0.366	0.351	0.308	0.316	0.327
	gen–snowy	0.385	0.371	0.332	0.341	0.343
	gen–sunset	0.386	0.366	0.285	0.325	0.335
	bl–autumn	0.369	0.365	0.308	0.328	0.326
	bl–snowy	0.373	0.362	0.305	0.320	0.319
	bl–sunset	0.360	0.353	0.285	0.328	0.316
FOG	baseline	0.332	**0.424**	0.305	0.367	**0.388**
	augment	**0.374**	0.382	0.334	**0.411**	0.361
	gen–autumn	0.364	0.358	0.326	0.397	0.352
	gen–snowy	0.346	0.363	**0.341**	0.403	0.351
	gen–aurora	0.289	0.313	0.313	0.358	0.282
	bl–autumn	0.304	0.320	0.300	0.365	0.288
	bl–snowy	0.316	0.333	0.340	0.382	0.316
	bl–aurora	0.305	0.319	0.334	0.376	0.287
SPRING	baseline	**0.385**	0.374	0.372	**0.451**	**0.392**
	augment	0.350	**0.376**	0.352	0.426	0.363
	gen–autumn	0.374	0.371	**0.384**	0.421	0.370
	gen–snowy	0.351	0.351	0.359	0.416	0.355
	gen–aurora	0.330	0.332	0.333	0.399	0.324
	bl–autumn	0.321	0.327	0.345	0.403	0.344
	bl–snowy	0.307	0.310	0.353	0.403	0.331
	bl–aurora	0.325	0.336	0.343	0.405	0.339

Table 7.8: *Training the augmented model on Highway (2)*

Generally the baseline performed better on most settings. The augmented model, trained on the original dataset seem to perform comparatively better than the models trained on the transformed datasets. On the domain most similar with the source (**HIGHWAY (2)**) there seems to be an improvement on the conditions most dissimilar with the train condition, but the impovements seem to be very condition–specific.

7.3 Results from additional training configurations on the augmented model

In the previous part we trained the models using the pipeline architecture suggested in [30]. In this part we apply different alterations to the 'augmented' pipeline and compare their performance:

- Configuration **A**: For the first 40 epochs of training , the original dataset is loaded to the augmented pipeline. For the remaining number of epochs we modify the pipeline by stopping the Contrastive Learning process and train the model using a transformed dataset.
- Configuration **B**: For each epoch, we train the Contrastive Module on the original dataset, and then we train the model on the transformed dataset.
- Configuration **C**: For each epoch, we train the Contrastive Module and apply the style augmentations to the original dataset, and then we train on a transformed dataset.
- Configuration **D**: For the first 40 epochs we train the augmented model on the original dataset, and then we train the model on a transformed dataset.

Several other configurations were tried, with performance significantly worse compared to the baseline. Below we show the combinations that show consistant improvement on one or more domains. As before, the trained models are compared with the unmodified 'baseline' model and the 'augment' model, trained on the original datasets. The title of each table describes which output of the DiffuseMix model we used, the textual prompt and the training domain.

An example of training loss and accuracy for Configuration A is provided in figure 7.3. Visual Diagrams of the configurations tested are provided in figures 7.4, 7.5, 7.6, 7.7. The performance of each modality is presented in the tables 7.9, 7.10, 7.11, 7.12.

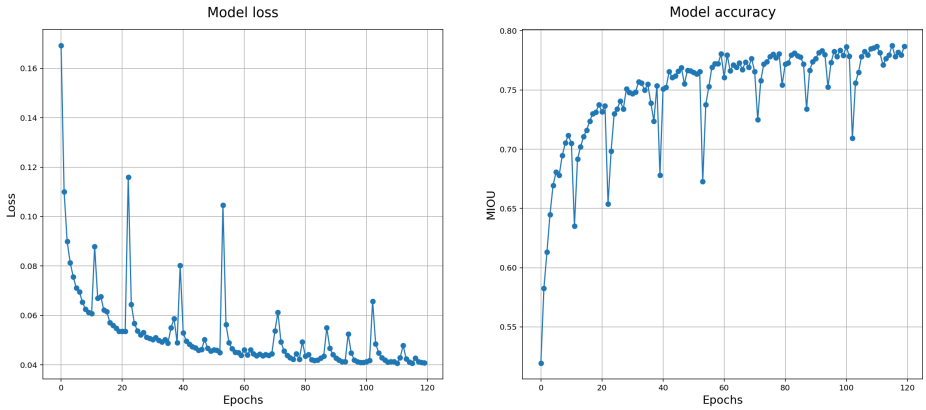


Image 7.3: *Diagrams of Loss and MIOU accuracy in training*

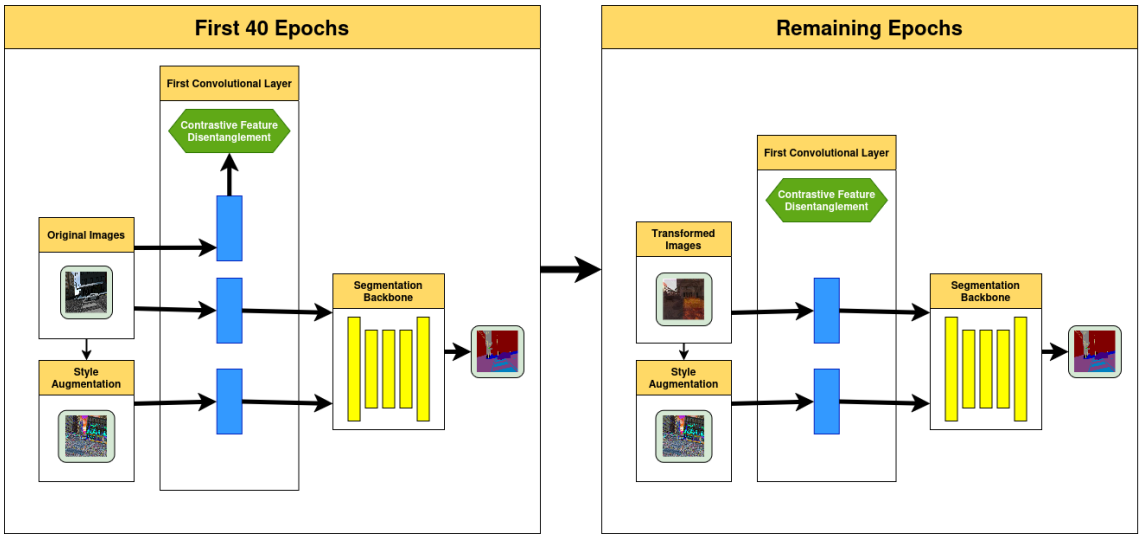


Image 7.4: *Diagram of the Configuration A*

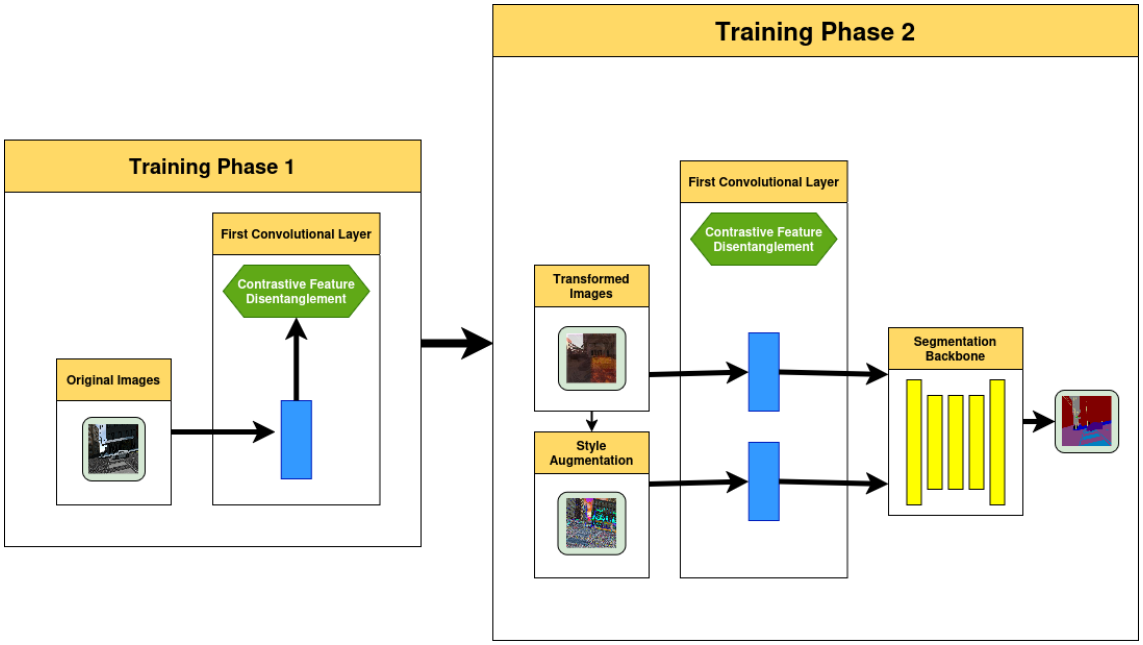


Image 7.5: *Diagram of the Configuration B*

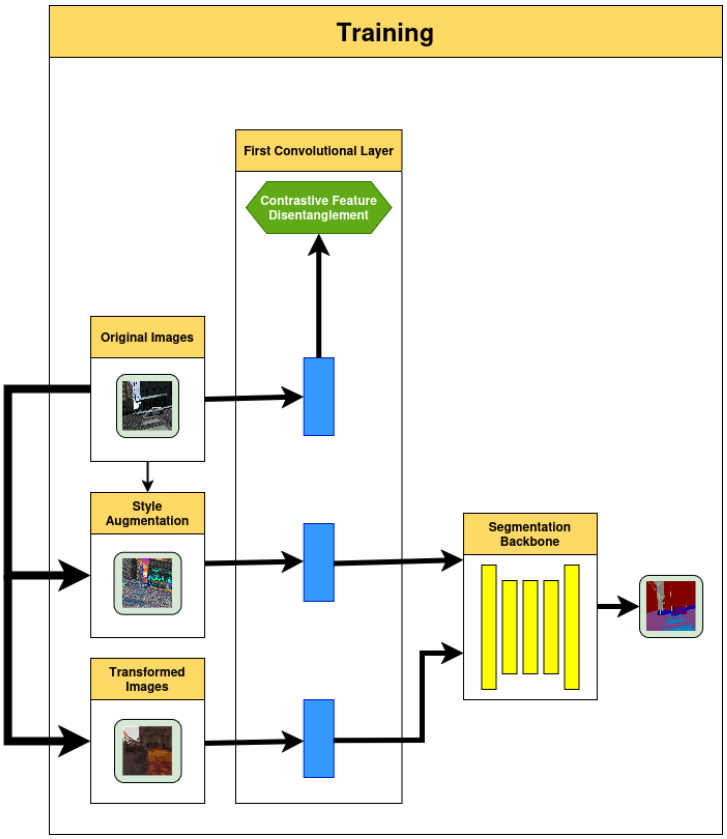


Image 7.6: *Diagram of the Configuration C*

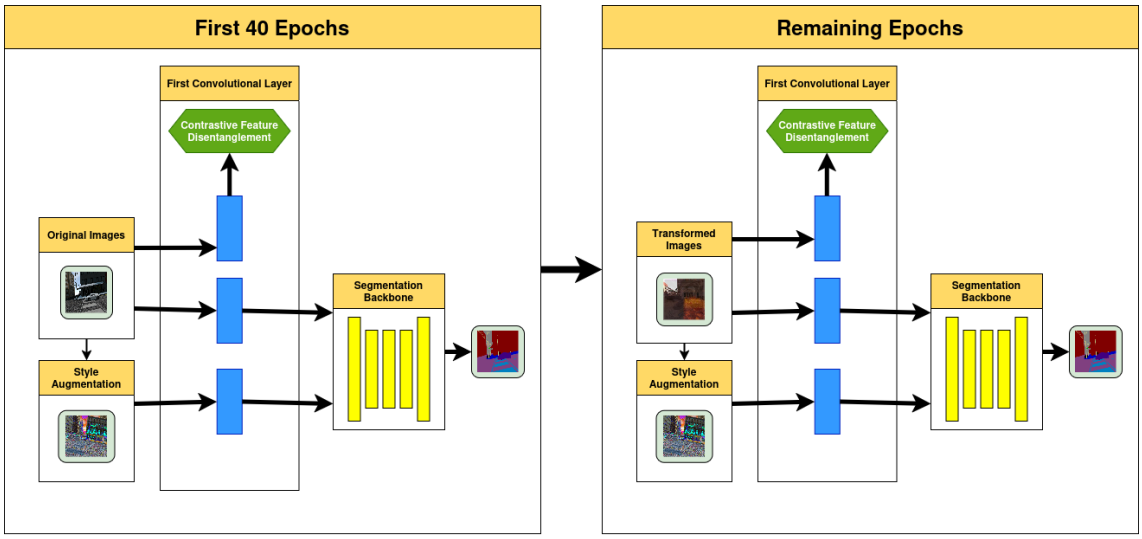


Image 7.7: *Diagram of the Configuration D*

generated,snowy,fog						
Test Domain	Mode	DAWN	FOG	NIGHT	SPRING	WINTER
New York City	baseline	**0.255**	**0.293**	**0.190**	**0.252**	**0.265**
	augment	0.186	0.224	0.131	0.188	0.176
	A	0.198	0.246	0.170	0.215	0.216
	B	0.159	0.217	0.129	0.180	0.176
	C	0.186	0.223	0.141	0.202	0.192
	D	0.154	0.206	0.112	0.171	0.169
Old European Town	baseline	**0.244**	**0.331**	**0.224**	**0.290**	**0.264**
	augment	0.227	0.242	0.190	0.230	0.193
	A	0.232	0.261	0.222	0.257	0.248
	B	0.188	0.199	0.163	0.201	0.210
	C	0.239	0.251	0.209	0.243	0.222
	D	0.182	0.186	0.147	0.180	0.187
New York City (2)	baseline	**0.316**	**0.358**	0.257	**0.358**	**0.289**
	augment	0.282	0.312	0.278	0.355	0.229
	A	0.296	0.336	**0.303**	0.350	0.278
	B	0.255	0.300	0.265	0.315	0.245
	C	0.264	0.313	0.272	0.336	0.242
	D	0.241	0.286	0.252	0.310	0.228
Highway (2)	baseline	0.332	**0.424**	0.305	0.367	**0.388**
	augment	**0.374**	0.382	0.334	0.411	0.361
	A	0.368	0.382	**0.368**	**0.414**	0.366
	B	0.357	0.370	0.353	0.405	0.360
	C	0.372	0.387	0.345	0.412	0.372
	D	0.344	0.369	0.340	0.400	0.357

Table 7.9: *Test fine-tuning approaches for the augmented model with generated images on the 'snowy' textual prompt. Training is done on the **FOG** condition*

blended,snowy,dawn						
Test Domain	Mode	DAWN	FOG	NIGHT	SPRING	WINTER
New York City	baseline	**0.305**	**0.282**	**0.249**	**0.288**	**0.268**
	augment	0.228	0.220	0.176	0.196	0.185
	A	0.225	0.232	0.197	0.209	0.194
	B	0.222	0.193	0.180	0.200	0.179
	C	0.223	0.205	0.171	0.192	0.172
	D	0.211	0.187	0.179	0.190	0.169
Old European Town	baseline	**0.281**	**0.329**	**0.304**	**0.346**	**0.269**
	augment	0.237	0.269	0.234	0.246	0.205
	A	0.211	0.242	0.217	0.238	0.211
	B	0.175	0.193	0.175	0.204	0.190
	C	0.188	0.215	0.189	0.210	0.177
	D	0.169	0.187	0.170	0.197	0.178
New York City (2)	baseline	**0.374**	**0.303**	**0.318**	**0.386**	**0.282**
	augment	0.318	0.272	0.242	0.300	0.198
	A	0.297	0.269	0.253	0.287	0.203
	B	0.266	0.233	0.223	0.268	0.190
	C	0.287	0.239	0.221	0.254	0.177
	D	0.258	0.224	0.224	0.250	0.183
Highway (2)	baseline	**0.415**	0.347	**0.356**	**0.389**	**0.386**
	augment	0.408	0.379	0.323	0.354	0.346
	A	0.390	**0.384**	0.328	0.350	0.345
	B	0.360	0.360	0.304	0.326	0.310
	C	0.389	0.380	0.326	0.336	0.337
	D	0.370	0.368	0.302	0.333	0.316

Table 7.10: *Test fine-tuning approaches for the augmented model with generated images on the 'snowy' textual prompt. Training is done on the **DAWN** condition*

ukiyo-e,fog						
Test Domain	Mode	DAWN	FOG	NIGHT	SPRING	WINTER
New York City	baseline	**0.255**	**0.293**	**0.190**	**0.252**	**0.265**
	augment	0.186	0.224	0.131	0.188	0.176
	bl−A	0.174	0.237	0.129	0.204	0.189
	gen−A	0.202	0.234	0.161	0.219	0.197
	bl−D	0.192	0.226	0.148	0.205	0.179
	gen−D	0.199	0.236	0.153	0.213	0.211
Old European Town	baseline	**0.244**	**0.331**	**0.224**	**0.290**	**0.264**
	augment	0.227	0.242	0.190	0.230	0.193
	bl−A	0.195	0.206	0.172	0.203	0.180
	gen−A	0.227	0.240	0.210	0.249	0.211
	bl−D	0.207	0.218	0.179	0.215	0.180
	gen−D	0.240	0.255	0.222	0.256	0.219
New York City (2)	baseline	**0.316**	**0.358**	0.257	**0.358**	**0.289**
	augment	0.282	0.312	0.278	0.355	0.229
	bl−A	0.248	0.303	0.265	0.314	0.236
	gen−A	0.277	0.307	**0.294**	0.351	0.243
	bl−D	0.252	0.303	0.267	0.320	0.230
	gen−D	0.283	0.308	0.285	0.347	0.247
Highway(2)	baseline	0.332	**0.424**	0.305	0.367	**0.388**
	augment	**0.374**	0.382	0.334	0.411	0.361
	bl−A	0.341	0.359	0.350	0.394	0.341
	gen−A	0.372	0.376	0.363	0.410	0.370
	bl−D	0.331	0.357	0.361	0.390	0.334
	gen−D	0.369	0.388	**0.369**	**0.414**	0.370

Table 7.11: *Test configurations A and D with generated and blended images on the 'ukiyo-e' textual prompt. Training is done on the **FOG** condition*

ukiyo-e,spring						
Test Domain	Mode	DAWN	FOG	NIGHT	SPRING	WINTER
New York City	baseline	**0.246**	**0.254**	**0.200**	**0.265**	0.252
	augment	0.227	0.249	0.190	0.262	**0.255**
	bl–A	0.197	0.218	0.169	0.224	0.237
	gen–A	0.206	0.219	0.186	0.218	0.238
	bl–D	0.195	0.226	0.163	0.234	0.243
	gen–D	0.203	0.233	0.184	0.214	0.239
Old European Town	baseline	**0.286**	**0.289**	**0.263**	**0.309**	0.266
	augment	0.262	0.257	0.226	0.267	**0.272**
	bl–A	0.227	0.198	0.186	0.218	0.213
	gen–A	0.236	0.222	0.214	0.231	0.240
	bl–D	0.236	0.196	0.178	0.211	0.218
	gen–D	0.246	0.228	0.215	0.240	0.245
New York City (2)	baseline	**0.325**	0.314	**0.311**	0.381	0.299
	augment	0.304	**0.330**	0.294	**0.385**	**0.313**
	bl–A	0.273	0.284	0.284	0.326	0.285
	gen–A	0.301	0.301	0.304	0.351	0.306
	bl–D	0.276	0.290	0.286	0.341	0.291
	gen–D	0.297	0.294	0.300	0.343	0.292
Highway (2)	baseline	**0.385**	0.374	0.372	**0.451**	**0.392**
	augment	0.350	**0.376**	0.352	0.426	0.363
	bl–A	0.346	0.347	0.380	0.417	0.346
	gen–A	0.366	0.365	0.384	0.418	0.367
	bl–D	0.339	0.341	0.370	0.405	0.346
	gen–D	0.372	0.364	**0.394**	0.420	0.364

Table 7.12: *Test configurations A and D with generated and blended images on the 'ukiyo-e' textual prompt. Training is done on the* ***SPRING*** *condition*

We can observe that generally these configurations perform similarly or worse than the base model or even the augmented model, with the exception of some gains occuring mainly with the **Highway (2)** environment and the NIGHT condition. The best performing configuration is A, with D also excibiting some improvement. It is also observed that the training on the images generated by the diffusion model provides better results than the training on the blended images.

7.4 Further investigation of the Highway (2) environment

We observed that testing on the **Highway (2)** had more success, comparing with the other environments. We present an overview across all training domains and modalities to find possible patterns.

The performance of each modality is presented in the tables 7.13, 7.14, 7.15.

Train Domain: Dawn						
Mode	Prompt	DAWN	FOG	NIGHT	SPRING	WINTER
baseline	-	**0.415**	0.347	**0.356**	**0.389**	**0.386**
augment	-	0.408	0.379	0.323	0.354	0.346
base-bl	autumn	0.384	0.365	0.277	0.347	0.339
base-gen	autumn	0.398	0.388	0.355	0.355	0.369
base-bl	ukiyo-e	0.396	0.384	0.329	0.346	0.347
base-gen	ukiyo-e	0.402	0.330	0.337	0.344	0.359
augment-bl	autumn	0.369	0.365	0.308	0.328	0.326
augment-gen	autumn	0.366	0.351	0.308	0.316	0.327
augment-bl	snowy	0.373	0.362	0.305	0.320	0.319
augment-gen	snowy	0.385	0.371	0.332	0.341	0.343
bl-A	autumn	0.385	0.374	0.336	0.350	0.339
gen-A	autumn	0.395	0.379	0.342	0.348	0.354
bl-A	snowy	0.390	0.384	0.328	0.350	0.345
gen-A	snowy	0.387	0.376	0.342	0.348	0.343
bl-A	ukiyo-e	0.388	0.378	0.316	0.323	0.340
gen-A	ukiyo-e	0.392	0.363	0.333	0.332	0.349
bl-B	autumn	0.360	0.349	0.291	0.318	0.331
gen-B	autumn	0.386	0.379	0.337	0.337	0.350
bl-B	snowy	0.360	0.360	0.304	0.326	0.310
gen-B	snowy	0.402	0.374	0.329	0.347	0.356
bl-C	autumn	0.397	0.377	0.318	0.342	0.349
bl-C	ukiyo-e	0.412	**0.394**	0.340	0.356	0.347
bl-D	snowy	0.370	0.368	0.302	0.333	0.316
gen-D	snowy	0.386	0.378	0.342	0.341	0.341
bl-D	ukiyo-e	0.398	0.389	0.342	0.351	0.349
gen-D	ukiyo-e	0.394	0.372	0.346	0.341	0.345

Table 7.13: *Outline of the performance of all methods, applied to environment* ***Highway (2)*** *with training condition* ***DAWN***

Train Domain: Fog						
Mode	Prompt	DAWN	FOG	NIGHT	SPRING	WINTER
baseline	-	0.332	**0.424**	0.305	0.367	**0.388**
augment	-	**0.374**	0.382	0.334	0.411	0.361
base-bl	autumn	0.330	0.367	0.261	0.382	0.337
base-gen	autumn	0.290	0.389	0.271	0.337	0.358
base-bl	ukiyo-e	0.322	0.365	0.291	0.387	0.333
base-gen	ukiyo-e	0.363	0.398	0.287	0.387	0.370
augment-bl	autumn	0.304	0.320	0.300	0.365	0.288
augment-gen	autumn	0.364	0.358	0.326	0.397	0.352
augment-bl	snowy	0.316	0.333	0.340	0.382	0.316
augment-gen	snowy	0.346	0.363	0.341	0.403	0.351
bl-A	aurora	0.338	0.360	0.345	0.391	0.331
gen-A	aurora	0.338	0.362	0.330	0.387	0.336
bl-A	snowy	0.346	0.366	0.358	0.403	0.350
gen-A	snowy	0.368	0.382	0.368	**0.414**	0.366
bl-A	ukiyo-e	0.341	0.359	0.350	0.394	0.341
gen-A	ukiyo-e	0.372	0.376	0.363	0.410	0.370
gen-B	snowy	0.357	0.370	0.353	0.405	0.360
gen-B	ukiyo-e	0.363	0.385	0.365	0.411	0.368
gen-C	snowy	0.372	0.387	0.345	0.412	0.372
bl-D	snowy	0.316	0.331	0.345	0.390	0.326
gen-D	snowy	0.344	0.369	0.340	0.400	0.357
bl-D	ukiyo-e	0.331	0.357	0.361	0.390	0.334
gen-D	ukiyo-e	0.369	0.388	**0.369**	**0.414**	0.370

Table 7.14: *Outline of the performance of all methods, applied to environment **Highway (2)** with training condition **FOG***

Train Domain: Spring						
Mode	Prompt	DAWN	FOG	NIGHT	SPRING	WINTER
baseline	-	**0.385**	0.374	0.372	**0.451**	**0.392**
augment	-	0.350	**0.376**	0.352	0.426	0.363
base-bl	autumn	0.324	0.325	0.347	0.417	0.343
base-gen	autumn	0.369	0.357	0.368	0.429	0.365
augment-bl	aurora	0.325	0.336	0.343	0.405	0.339
augment-gen	aurora	0.330	0.332	0.333	0.399	0.324
augment-bl	autumn	0.321	0.327	0.345	0.403	0.344
augment-gen	autumn	0.374	0.371	0.384	0.421	0.370
augment-bl	snowy	0.307	0.310	0.353	0.403	0.331
augment-gen	snowy	0.351	0.351	0.359	0.416	0.355
bl-A	aurora	0.357	0.362	0.363	0.424	0.361
gen-A	aurora	0.371	0.365	0.355	0.420	0.367
bl-A	autumn	0.374	0.370	0.384	0.428	0.372
gen-A	autumn	0.383	0.372	**0.398**	0.435	0.383
bl-A	ukiyo-e	0.346	0.347	0.380	0.417	0.346
gen-A	ukiyo-e	0.366	0.365	0.384	0.418	0.367
bl-B	ukiyo-e	0.342	0.347	0.365	0.408	0.344
gen-C	snowy	0.367	0.369	0.372	0.435	0.366
bl-D	ukiyo-e	0.339	0.341	0.370	0.405	0.346
gen-D	ukiyo-e	0.372	0.364	0.394	0.420	0.364

Table 7.15: *Outline of the performance of all methods, applied to environment **Highway (2)** with training condition **SPRING***

We can see more clearly that for similar training environments, the methods we use can outperform the baseline, in particular in conditions that are distant from the training domain, such as **FOG** with the training done on **DAWN** or **FOG** and **NIGHT** with training done on **SPRING**. For training in **FOG** our method seem to improve the performance by a significant margin in most of the distant domains, such as **DAWN** and **SPRING**.

Chapter 8

Conclusion

8.1 Conclusion

In this thesis we showed that the use of our augmented pipeline can improve the baseline in some cases. When testing our trained model on the most similar domain, we observed performance improvement for conditions that are dissimilar with the training setting. We can conclude that our design can be useful for fine-tuning a model to adverse domains. We also observed that the selection of the textual prompt influenced the performance of the model. For the same setting, there were one or more textual prompts that performed significantly better, comparing with other prompts. The best performing prompts also differed between domains, with each training condition suggesting a different subset of optimal prompts.

Nonetheless, in most cases there was no improvement in performance, with baseline performing the best by a significant margin. This can be attributed to insufficient regularization, or other hyperparameter settings which caused the model to be unable to generalize, issues that despite extensive experimentation were not addressed. It can be suggested that this task cannot be sufficiently addressed by our architectural choices, but the occasional improvements should be further investigated.

It must be noted that the CCSDG model was developed for a semantic segmentation task, while the segmentation tasks with the SYNTHIA dataset may be addressed more efficiently in an instance segmentation setting.

8.2 Future Work

There are many paths to build on this work. We can expand on the utilization of the DiffuseMix pipeline, with the combination of images produced with multiple textual prompts in one training dataset, with a possibility of an optimal conbination of these prompts. For an easier route, we can introduce another model architecture, for example an instance segmentation model or a ViT, or expand the capabilities of our FCN model. It will be beneficial to apply this architecture on a classic Domain Generalization task such as the SYNTHIA→Cityscapes that can help us reveal the pattern behind the increases in performance. An in-depth semantic class analysis, and a possible introduction of a class discriminatory training scheme might prove to be of use.

Βιβλιογραφία

[1] Marius Cordts, Mohamed Omran, Sebastian Ramos, Timo Rehfeld, Markus Enzweiler, Rodrigo Benenson, Uwe Franke, Stefan Roth και Bernt Schiele. *The Cityscapes Dataset for Semantic Urban Scene Understanding*, 2016.

[2] Jonathan Long, Evan Shelhamer και Trevor Darrell. *Fully Convolutional Networks for Semantic Segmentation*, 2015.

[3] Kaiming He, Georgia Gkioxari, Piotr Dollár και Ross Girshick. *Mask R-CNN*, 2018.

[4] *Overview of a Neural Network's Learning Process*. https://medium.com/data-science-365/overview-of-a-neural-networks-learning-process-61690a502fa. Ημερομηνία πρόσβασης: 14-06-2025.

[5] *Model Fit: Underfitting vs. Overfitting*. https://docs.aws.amazon.com/machine-learning/latest/dg/model-fit-underfitting-vs-overfitting.html. Ημερομηνία πρόσβασης: 19-06-2025.

[6] Keiron O'Shea και Ryan Nash. *An Introduction to Convolutional Neural Networks*, 2015.

[7] *Convolution operation*. https://www.researchgate.net/figure/Convolution-operation_fig2_355656417. Ημερομηνία πρόσβασης: 22-06-2025.

[8] *Illustration-of-Max-Pooling-and-Average-Pooling*. https://www.researchgate.net/figure/llustration-of-Max-Pooling-and-Average-Pooling-Figure-2-above-shows-an-example-of-max_fig2_333593451. Ημερομηνία πρόσβασης: 22-06-2025.

[9] Jonathan Ho, Ajay Jain και Pieter Abbeel. *Denoising Diffusion Probabilistic Models*, 2020.

[10] *Visualization of how Contrastive Learning works*. https://www.researchgate.net/figure/sualization-of-how-Contrastive-Learning-works-The-process-involves-augmenting-the_fig3_382262406. Ημερομηνία πρόσβασης: 25-06-2025.

[11] German Ros, Laura Sellart, Joanna Materzynska, David Vazquez, και Antonio M. Lopez. *The SYNTHIA Dataset: A Large Collection of Synthetic Images for Semantic Segmentation of Urban Scenes*. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Las Vegas, NV, USA, 2016.

[12] Kaiyang Zhou, Ziwei Liu, Yu Qiao, Tao Xiang και Chen Change Loy. *Domain Generalization: A Survey*. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, σελίδα 1–20, 2022.

[13] Tomer Amit, Tal Shaharbany, Eliya Nachmani και Lior Wolf. *SegDiff: Image Segmentation with Diffusion Probabilistic Models*, 2022.

[14] Yanchao Yang και Stefano Soatto. *FDA: Fourier Domain Adaptation for Semantic Segmentation*, 2020.

[15] Shervin Minaee, Yuri Boykov, Fatih Porikli, Antonio Plaza, Nasser Kehtarnavaz και Demetri Terzopoulos. *Image Segmentation Using Deep Learning: A Survey*, 2020.

[16] Risheng Wang, Tao Lei, Ruixia Cui, Bingtao Zhang, Hongying Meng και Asoke K. Nandi. *Medical image segmentation using deep learning: A survey*. *IET Image Processing*, 16(5):1243–1267, 2022.

[17] Senay Cakir, Marcel Gauß, Kai Häppeler, Yassine Ounajjar, Fabian Heinle και Reiner Marchthaler. *Semantic Segmentation for Autonomous Driving: Model Evaluation, Dataset Generation, Perspective Comparison, and Real-Time Capability*, 2022.

[18] Zewen Li, Wenjie Yang, Shouheng Peng και Fan Liu. *A Survey of Convolutional Neural Networks: Analysis, Applications, and Prospects*, 2020.

[19] Sergey Ioffe και Christian Szegedy. *Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift*, 2015.

[20] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever και Ruslan Salakhutdinov. *Dropout: A Simple Way to Prevent Neural Networks from Overfitting*. *Journal of Machine Learning Research*, 15(56):1929–1958, 2014.

[21] Kaiming He, Xiangyu Zhang, Shaoqing Ren και Jian Sun. *Deep Residual Learning for Image Recognition*, 2015.

[22] Jascha Sohl-Dickstein, Eric A. Weiss, Niru Maheswaranathan και Surya Ganguli. *Deep Unsupervised Learning using Nonequilibrium Thermodynamics*, 2015.

[23] R. Hadsell, S. Chopra και Y. LeCun. *Dimensionality Reduction by Learning an Invariant Mapping*. *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06)*, τόμος 2, σελίδες 1735–1742, 2006.

[24] Florian Schroff, Dmitry Kalenichenko και James Philbin. *FaceNet: A unified embedding for face recognition and clustering*. *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, σελίδα 815–823. IEEE, 2015.

[25] Xinlei Chen, Haoqi Fan, Ross Girshick και Kaiming He. *Improved Baselines with Momentum Contrastive Learning*, 2020.

[26] *Full Guide to Contrastive Learning*. https://encord.com/blog/guide-to-contrastive-learning/. Ημερομηνία πρόσβασης: 24-06-2025.

[27] Hao Guan και Mingxia Liu. *Domain Adaptation for Medical Image Analysis: A Survey*. *IEEE Transactions on Biomedical Engineering*, 69(3):1173–1185, 2022.

[28] Zixian Su, Kai Yao, Xi Yang, Qiufeng Wang, Jie Sun και Kaizhu Huang. *Rethinking Data Augmentation for Single-source Domain Generalization in Medical Image Segmentation*, 2022.

[29] Mortenson. *Mathematics for computer graphics applications*. Industrial Press Inc., 1999.

[30] Shishuai Hu, Zehui Liao και Yong Xia. *Devil is in Channels: Contrastive Single Domain Generalization for Medical Image Segmentation*, 2023.

[31] Ling Zhang, Xiaosong Wang, Dong Yang, Thomas Sanford, Stephanie Harmon, Baris Turkbey, Bradford J. Wood, Holger Roth, Andriy Myronenko, Daguang Xu και Ziyue Xu. *Generalizing Deep Learning for Medical Image Segmentation to Unseen Domains via Deep Stacked Transformation*. *IEEE Transactions on Medical Imaging*, 39(7):2531–2540, 2020.

[32] Tim Brooks, Aleksander Holynski και Alexei A. Efros. *InstructPix2Pix: Learning to Follow Image Editing Instructions*, 2023.

[33] Khawar Islam, Muhammad Zaigham Zaheer, Arif Mahmood και Karthik Nandakumar. *DiffuseMix: Label-Preserving Data Augmentation with Diffusion Models*, 2024.

[34] *L1 loss, by torch.nn.functional*. https://docs.pytorch.org/docs/stable/generated/torch.nn.functional.l1_loss.html. Ημερομηνία πρόσβασης: 07-06-2025.

[35] *Cross Entropy Loss*. https://docs.pytorch.org/docs/stable/generated/torch.nn.CrossEntropyLoss.html. Ημερομηνία πρόσβασης: 26-05-2025.

[36] *Multiclass Jaccard Index*. https://lightning.ai/docs/torchmetrics/stable/classification/jaccard_index.html. Ημερομηνία πρόσβασης: 24-05-2025.

[37] Alex Krizhevsky, Ilya Sutskever και Geoffrey E. Hinton. *ImageNet classification with deep convolutional neural networks*. *Commun. ACM*, 60(6):84–90, 2017.

[38] Karen Simonyan και Andrew Zisserman. *Very Deep Convolutional Networks for Large-Scale Image Recognition*, 2015.

[39] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke και Andrew Rabinovich. *Going Deeper with Convolutions*, 2014.

[40] Saeid Motiian, Marco Piccirilli, Donald A. Adjeroh και Gianfranco Doretto. *Unified Deep Supervised Domain Adaptation and Generalization*, 2017.

[41] Divyat Mahajan, Shruti Tople και Amit Sharma. *Domain Generalization using Causal Matching*, 2021.

[42] Shoubo Hu, Kun Zhang, Zhitang Chen και Laiwan Chan. *Domain Generalization via Multidomain Discriminant Analysis*, 2019.

[43] Da Li, Yongxin Yang, Yi Zhe Song και Timothy M. Hospedales. *Learning to Generalize: Meta-Learning for Domain Generalization*, 2017.

[44] Yogesh Balaji, Swami Sankaranarayanan και Rama Chellappa. *MetaReg: Towards Domain Generalization using Meta-Regularization*. *Advances in Neural Information Processing Systems*S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi και R. Garnett, επιμελητές, τόμος 31. Curran Associates, Inc., 2018.

[45] Qi Dou, Daniel C. Castro, Konstantinos Kamnitsas και Ben Glocker. *Domain Generalization via Model-Agnostic Learning of Semantic Features*, 2019.

[46] Riccardo Volpi και Vittorio Murino. *Addressing Model Vulnerability to Distributional Shifts over Image Transformation Sets*, 2019.

[47] Aman Sinha, Hongseok Namkoong, Riccardo Volpi και John Duchi. *Certifying Some Distributional Robustness with Principled Adversarial Training*, 2020.

[48] Kaiyang Zhou, Yongxin Yang, Timothy Hospedales και Tao Xiang. *Deep Domain-Adversarial Image Generation for Domain Generalisation*, 2020.

[49] Kaiyang Zhou, Yongxin Yang, Yu Qiao και Tao Xiang. *Domain Adaptive Ensemble Learning*. *IEEE Transactions on Image Processing*, 30:8008–8018, 2021.

[50] Antonio D'Innocente και Barbara Caputo. *Domain Generalization with Domain-Specific Aggregation Modules*, 2018.

[51] Fabio Maria Carlucci, Antonio D'Innocente, Silvia Bucci, Barbara Caputo και Tatiana Tommasi. *Domain Generalization by Solving Jigsaw Puzzles*, 2019.

[52] Da Li, Yongxin Yang, Yi Zhe Song και Timothy M. Hospedales. *Deeper, Broader and Artier Domain Generalization*, 2017.

[53] Haohan Wang, Zexue He, Zachary C. Lipton και Eric P. Xing. *Learning Robust Representations by Projecting Superficial Statistics Out*, 2019.

[54] Kaiyang Zhou, Yongxin Yang, Yu Qiao και Tao Xiang. *Domain Generalization with MixStyle*, 2021.

[55] Michael Laskin, Kimin Lee, Adam Stooke, Lerrel Pinto, Pieter Abbeel και Aravind Srinivas. *Reinforcement Learning with Augmented Data*, 2020.

[56] Nicklas Hansen και Xiaolong Wang. *Generalization in Reinforcement Learning by Soft Data Augmentation*, 2021.

[57] *Domain shift*. https://www.statlect.com/machine-learning/domain-shift. Ημερομηνία πρόσβασης: 21-06-2025.

[58] Stephan R. Richter, Vibhav Vineet, Stefan Roth και Vladlen Koltun. *Playing for Data: Ground Truth from Computer Games*, 2016.

[59] Wilhelm Tranheden, Viktor Olsson, Juliano Pinto και Lennart Svensson. *DACS: Domain Adaptation via Cross-domain Mixed Sampling*, 2020.

[60] Tuan Hung Vu, Himalaya Jain, Maxime Bucher, Matthieu Cord και Patrick Pérez. *DADA: Depth-aware Domain Adaptation in Semantic Segmentation*, 2019.

[61] Qianyu Zhou, Zhengyang Feng, Qiqi Gu, Jiangmiao Pang, Guangliang Cheng, Xuequan Lu, Jianping Shi και Lizhuang Ma. *Context-Aware Mixup for Domain Adaptive Semantic Segmentation*. *IEEE Transactions on Circuits and Systems for Video Technology*, 33(2):804–817, 2023.

[62] Yi Hsuan Tsai, Kihyuk Sohn, Samuel Schulter και Manmohan Chandraker. *Domain Adaptation for Structured Output via Discriminative Patch Representations*, 2019.

[63] Luke Melas-Kyriazi και Arjun K. Manrai. *PixMatch: Unsupervised Domain Adaptation via Pixelwise Consistency Training*, 2021.

[64] Yuhua Chen, Wen Li και Luc Van Gool. *ROAD: Reality Oriented Adaptation for Semantic Segmentation of Urban Scenes*, 2018.

[65] Judy Hoffman, Eric Tzeng, Taesung Park, Jun Yan Zhu, Phillip Isola, Kate Saenko, Alexei A. Efros και Trevor Darrell. *CyCADA: Cycle-Consistent Adversarial Domain Adaptation*, 2017.

[66] Zhonghao Wang, Mo Yu, Yunchao Wei, Rogerio Feris, Jinjun Xiong, Wenmei Hwu, Thomas S. Huang και Humphrey Shi. *Differential Treatment for Stuff and Things: A Simple Unsupervised Domain Adaptation Method for Semantic Segmentation*, 2020.

[67] Shaoan Xie, Zibin Zheng, Liang Chen και Chuan Chen. *Learning Semantic Representations for Unsupervised Domain Adaptation*. *Proceedings of the 35th International Conference on Machine Learning*Jennifer Dy και Andreas Krause, επιμελητές, τόμος 80 στο *Proceedings of Machine Learning Research*, σελίδες 5423–5432. PMLR, 2018.

[68] Yawei Luo, Liang Zheng, Tao Guan, Junqing Yu και Yi Yang. *Taking A Closer Look at Domain Shift: Category-level Adversaries for Semantics Consistent Domain Adaptation*, 2019.

[69] Yang Zou, Zhiding Yu, B. V. K. Vijaya Kumar και Jinsong Wang. *Domain Adaptation for Semantic Segmentation via Class-Balanced Self-Training*, 2018.

[70] Rui Gong, Wen Li, Yuhua Chen και Luc Van Gool. *DLOW: Domain Flow for Adaptation and Generalization*, 2019.

[71] Kaiyang Zhou, Yongxin Yang, Timothy Hospedales και Tao Xiang. *Learning to Generate Novel Domains for Domain Generalization*, 2021.

[72] Riccardo Volpi, Hongseok Namkoong, Ozan Sener, John Duchi, Vittorio Murino και Silvio Savarese. *Generalizing to Unseen Domains via Adversarial Data Augmentation*, 2018.

[73] Dan Hendrycks, Andy Zou, Mantas Mazeika, Leonard Tang, Bo Li, Dawn Song και Jacob Steinhardt. *PixMix: Dreamlike Pictures Comprehensively Improve Safety Measures*, 2022.

[74] Sangdoo Yun, Dongyoon Han, Seong Joon Oh, Sanghyuk Chun, Junsuk Choe και Youngjoon Yoo. *CutMix: Regularization Strategy to Train Strong Classifiers with Localizable Features*, 2019.

[75] A. F. M. Shahab Uddin, Mst. Sirazam Monira, Wheemyung Shin, TaeChoong Chung και Sung Ho Bae. *SaliencyMix: A Saliency Guided Data Augmentation Strategy for Better Regularization*, 2021.

[76] Shaoli Huang, Xinchao Wang και Dacheng Tao. *SnapMix: Semantically Proportional Mixing for Augmenting Fine-grained Data*, 2020.

[77] Chen Chen, Zeju Li, Cheng Ouyang, Matt Sinclair, Wenjia Bai και Daniel Rueckert. *MaxStyle: Adversarial Style Composition for Robust Medical Image Segmentation*, 2022.

[78] Chitwan Saharia, William Chan, Saurabh Saxena, Lala Li, Jay Whang, Emily Denton, Seyed Kamyar Seyed Ghasemipour, Burcu Karagol Ayan, S. Sara Mahdavi, Rapha Gontijo Lopes, Tim Salimans, Jonathan Ho, David J Fleet και Mohammad Norouzi. *Photorealistic Text-to-Image Diffusion Models with Deep Language Understanding*, 2022.

[79] Prafulla Dhariwal και Alex Nichol. *Diffusion Models Beat GANs on Image Synthesis*, 2021.

[80] Brandon Trabucco, Kyle Doherty, Max Gurinas και Ruslan Salakhutdinov. *Effective Data Augmentation With Diffusion Models*, 2023.

[81] Shekoofeh Azizi, Simon Kornblith, Chitwan Saharia, Mohammad Norouzi και David J. Fleet. *Synthetic Data from Diffusion Models Improves ImageNet Classification*, 2023.

[82] Weijia Wu, Yuzhong Zhao, Mike Zheng Shou, Hong Zhou και Chunhua Shen. *DiffuMask: Synthesizing Images with Pixel-level Annotations for Semantic Segmentation Using Diffusion Models*, 2024.

[83] Dmitry Baranchuk, Ivan Rubachev, Andrey Voynov, Valentin Khrulkov και Artem Babenko. *Label-Efficient Semantic Segmentation with Diffusion Models*, 2022.

[84] Julia Wolleb, Florentin Bieder, Robin Sandkühler και Philippe C. Cattin. *Diffusion Models for Medical Anomaly Detection*, 2022.

[85] Pedro Sanchez, Antanas Kascenas, Xiao Liu, Alison Q. O'Neil και Sotirios A. Tsaftaris. *What is Healthy? Generative Counterfactual Diffusion for Lesion Localization*, 2022.