Міністерство освіти і науки України

Харківський національний університет радіоелектроніки

Факультет _____Комп'ютерних наук_____
(повна назва)

Кафедра _____Програмної інженерії_____
(повна назва)

АТЕСТАЦІЙНА РОБОТА

Пояснювальна записка

рівень вищої освіти – другий (магістерський)

Дослідження методів комп'ютерного зору для адаптації домену попередньо натренованих моделей
(тема)

Виконав: студент 2 курсу, групи ІПЗм-18-3
_____Юсіфов Р.Г._____
(прізвище, ініціали)

____Освітньо-наукової програми___
(тип програми)

____Інженерія програмного забезпечення___
(повна назва освітньої програми)

Керівник _к.т.н доц. каф. ПІ Турута О.П._
(посада, прізвище, ініціали)

Допускається до захисту
Зав. Кафедри, проф. _____ З.В.Дудар
20__р.

ХАРКІВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ РАДІОЕЛЕКТРОНІКИ

Факультет _Комп'ютерних наук_____

Кафедра _Програмної інженерії_____

Рівень вищої освіти – другий (магістерський)

Спеціальність _121 – Інженерія програмного забезпечення_____
(код і повна назва)

Тип програми _освітньо-наукова програма_____

Освітня програма _Інженерія програмного забезпечення_____

ЗАТВЕРДЖУЮ:

Зав. кафедри _____
(підпис)
«____»_____20__ р.

ЗАВДАННЯ

НА АТЕСТАЦІЙНУ РОБОТУ

студентові _Юсіфову Рамалу Гейбатовичу_____
(прізвище, ім'я, по батькові)

1. Тема роботи _Дослідження методів комп'ютерного зору для адаптації домену попередньо натренованих моделей
затверджена наказом університету від "__"_____ 20__ р. № _____
заповнюється вручну після отримання наказу
2. Термін подання студентом роботи до екзаменаційної комісії
20 травня 2020 р.
3. Вихідні дані до роботи *алгоритми доменної адаптації для задач комп'ютерного зору, пояснювальна записка. Використовувати ОС Linux, бібліотеки глибинного навчання PyTorch*_____
4. Перелік питань, що потрібно опрацювати в роботі *мета роботи, аналіз проблемної галузі і постановка задачі, огляд нейронних мереж для семантичної сегментації*,

методи адаптації домену для задач комп'ютерного зору та конкретно семантичної сегментації зображень, методи балансування класів при адаптації домену.

5. Консультанти розділів роботи

Найменування розділу	Консультант (посада, прізвище, ім'я, по батькові)	Позначка консультанта про виконання розділу	
		підпис	дата
Спецчастина	к.т.н., доц. каф. ПІ Турута О.П.		

КАЛЕНДАРНИЙ ПЛАН

№	Назва етапів роботи	Термін виконання етапів роботи	Примітка
1.	Аналіз предметної галузі	7 квітня 2020 р.	
2.	Огляд існуючих методів адаптації домену	15 квітня 2020 р.	
3.	Методи семантичної сегментації	22 квітня 2020 р.	
4.	Підготовка пояснювальної записки	2 травня 2020 р.	
5.	Спецчастина	6 травня 2020 р.	
6.	Підготовка презентацій та доповіді	10 травня 2020 р.	
7.	Попередній захист	13 травня 2020 р.	
8.	Нормоконтроль, рецензування	15 травня 2020 р.	
9.	Занесення диплома в електронний архів	18 травня 2020 р.	
10.	Допуск до захисту у зав. кафедри	19 травня 2020 р.	

Дата видачі завдання ____ _____ 2020 р.

Студент _____
(підпис)

Керівник роботи _____ ____к.т.н., доц. каф. ПІ Турута О.П.__
(підпис) (посада, прізвище, ініціали)

РЕФЕРАТ / ABSTRACT

Звіт з науково-дослідної практики: 47 с., 12 рис., 3 табл., 40 джер.

АДАПТАЦІЯ ДОМЕНУ, ГЛИБИННЕ НАВЧАННЯ, МАШИННЕ НАВЧАННЯ, ЦІЛЬОВИЙ ДОМЕН, КОМП'ЮТЕРНИЙ ЗІР, КЛАСИФІКАЦІЯ ЗОБРАЖЕНЬ, СЕМАНТИЧНА СЕГМЕНТАЦІЯ, КОНВОЛЮЦІЙНА НЕЙРОННА МЕРЕЖА, КОНТРОЛЬОВАНЕ НАВЧАННЯ, ПСЕВДОМІТКА.

Метою роботи є дослідження методів доменної адаптації у галузі комп'ютерного зору, а саме розробка методології доменної адаптації для задач семантичної сегментації.

Методи розробки включають мову програмування Python та бібліотеку для глибинного навчання PyTorch.

В результаті роботи розглянуто існуючі методи доменної адаптації для семантичної сегментації зображень. Для методу адаптації домену шляхом генерації псевдоміток запропоновано вирішення проблеми балансування класів, при якому враховується схожий розподіл класів за пікселями між доменами. Запропонований підхід було перевірено при адаптації моделі, натренованій на датасеті GTA, до датасету Cityscapes.

DOMAIN ADAPTATION, DEEP LEARNING, MACHINE LEARNING, TARGET DOMAIN, COMPUTER VISION, IMAGE CLASSIFICATION, SEMANTIC SEGMENTATION, CONVOLUTIONAL NEURAL NETWORK, SUPERVISED TRAINING, PSEUDO-LABEL.

The aim of this work is to study the methods of domain adaptation in the field of computer vision, specifically the development of a methodology for domain adaptation for semantic segmentation problems.

Development methods are based on the Python programming language and the PyTorch in-depth learning library.

The existing methods of domain adaptation for semantic segmentation are reviewed. For the method of pseudo-label generation, a solution to the problem of class balancing is proposed, which uses the similar distribution of classes by pixels between domains. The proposed method was examined on the adaptation of the model trained on the GTA dataset to the Cityscapes dataset.

ЗМІСТ

ВСТУП

Глибинне навчання – галузь машинного навчання, алгоритми якої намагаються моделювати високорівневі залежності між даними будуючи глибинний граф, що містить лінійні та нелінійні перетворення[1]. Для деяких задач вже розроблені інтелектуальні системи, ефективність яких можна порівнювати з людиною. Так наприклад, системи з розпізнавання предметів на зображеннях вже здатні розпізнавати об’єкти краще за людей[2]. Такі результати були досягнути завдяки здатності систем глибинного навчання обробляти велику кількість початкових даних. Глибинне навчання стало популярним завдяки масштабному збору даних та відкритому доступу до наборів даних. Алгоритми навчання тепер вбудовані в системи керування автомобілями[3], наведення безпілотників, комп'ютерної діагностики, інтернет-торгівлі, супутникова картографії, тощо.

Незважаючи на велику популярність глибинного навчання у наш час, все одно існує велика нестача розмічених даних. Крім того, традиційне глибинне навчання не дає змогу моделі, навченій на одних даних переключатися на інші дані. Отже, постає проблема – яким чином застосувати вже створену модель для дещо інших даних або як навчити модель при відсутності достатньої кількості розмічених даних. Доменна адаптація займається рішенням цієї проблеми.

В роботі запропоновано спосіб балансування класів при доменній адаптації з генерацією псевдоміток. Запропонований метод використовує дані про схожий розподіл класів між доменами для фільтрації псевдоміток і балансування класів. Дослідження проводилися на адаптації моделі, навченої на датасеті GTA, до датасету Cityscapes. Для визначення ефективності методу результати порівнювали на моделі, навченої на вихідному датасеті, та моделі, навченої на цільовому датасеті. Розглянуто шляхи поліпшення методу, подальші дослідження і перспективи застосування.

1 АНАЛІЗ ПРЕДМЕТНОЇ ГАЛУЗІ

1.1 Обґрунтування актуальності обраної теми

У зв’язку з появою великих датасетів та вдосконаленням апаратних засобів, глибинне навчання стало головним підходом до вирішення завдань штучного інтелекту. Тільки у галузі комп’ютерного зору завдяки глибинному навчанню були досягнуті високі результати у класифікації зображень, виявленню об’єктів, семантичній сегментації, сегментації екземплярів, тощо.

Втім створення ефективних рішень з глибинним навчанням потребує навчання на великих обсягах даних, розмір деяких датасетів може перевищувати мільйона зразків. Отже, навчання моделей – ресурсномісткий процес, що потребує багато часу. Наприклад, процес навчання мовної моделі BERT, розробленої Google, зайняв чотирі дні на 16 тензорних процесорах[4].

Навіть при наявності великих датасетів, апаратних засобів та необхідного часу, навчені моделі можуть виконувати лише одну задачу, а навчання нової моделі для кожної окремої задачі не завжди можливо. По-перше, як це вже було показано вище, це дуже трудомісткий процес. По-друге, в багатьох задачах відсутні розмічені дані для навчання, або їхня кількість дуже мала. Альтернативний варіант вихід – налаштування вже готових моделей для інших задач – отримав назву трансферного навчання.

1.2 Основні поняття глибинного навчання

Оскільки в роботі порівнюються архітектури нейронних мереж, далі стисло наведено основні поняття глибинного навчання і принципи його роботи. В широкому сенсі методи глибинного навчання приймають на вхід дані X, обробляють ці дані та

видають Y. Нейронна мережа представляє собою великий обчислювальний граф. Вона складається з декількох шарів, кожний шар містить певну кількість вузлів. Інформація від шару до шару проходить завдяки функції активації (рисунок 1.1).

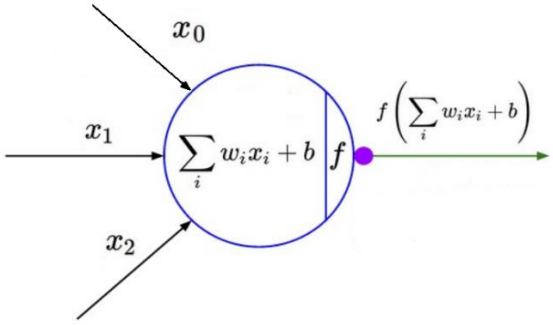

Рисунок 1.1 – Схема функції активації

Параметри функції визначають w та b результати вихідного шару. Мета процесу навчання – отримати такі w та b, при яких досягається висока точність моделі. Оцінка моделі розраховується як різниця між отриманими і очікуваними даними – функція втрат. Формула для середньоквадратичної помилки наведено нижче.

$$\mathrm{F} = \frac{1}{n}\sum_{i=1}^{n}(Y_i - \hat{Y}_i)^2 \qquad (1.1)$$

де $\hat{Y}$ – прогноз моделі, Y – очікуване значення, n – кількість зразків.

Сутність оптимізації полягає у мінімізації втрат через корегування вагових коефіцієнтів мережі. Зміна коефіцієнтів w та b виконується градієнтний спуск під час зворотного поширення помилки. З кожною ітерацією зворотного поширення помилки значення функції втрат зменшується. Параметри, що задаються перед початком оптимізації моделі для контролювання процесу навчання, отримали назву гіперпараметрів. Темп навчання контролює наскільки сильно змінюються вагові коефіцієнти між ітераціями, тобто впливає на швидкість навчання.

1.3 Формулювання задачі доменної адаптації

Домен D складається з d-мірного простору ознак $X \subset R^d$ з відособленим розподілом ймовірностей $P(X)$, та задачею τ, що визначається простором міток γ та розподілом умовних ймовірностей $P(Y|X)$, де X та Y – це випадкові величини. За умови, що дано деякий набір даних $\acute{X} = \{x_1, \dots, x_n\}$, усі дані, якого належать простору X, з відповідними мітками $\acute{Y} = \{y_1, \dots, y_n\}$ з простору γ, $P(Y|X)$ можна навчити шляхом контрольованого навчання за цими парами ознак та міток $\{x_i, y_i\}$[5].

Допустимо, що існують два домени, кожний з яких має свою задачу: вихідний домен $D^s = \{X^s, P(X^s)\}$ з задачею $\tau^s = \{y^s, P(Y^s|X^s)\}$ та цільовий домен $D^t = \{X^t, P(X^t)\}$ з задачею $\tau^t = \{y^t, P(Y^t|X^t)\}$. Якщо два домени співпадають, тобто $D^s = D^t$ та $\tau^s = \tau^t$, достатньо застосувати традиційні методи машинного навчання для вирішення поставленої проблеми. D^s обирається в якості навчального датасету, а D^t – в якості тестового.

Якщо два домени не співпадають, тобто $D^t \neq D^s$ або $\tau^t \neq \tau^s$, моделі, натреновані на D^s можуть демонструвати погані результати на домені D^t, або модель не можна застосовувати на домені D^t якщо $\tau^t \neq \tau^s$. Коли вихідний домен має певну схожість з цільовим доменом, існує можливість використати дані з $\{D^s, \tau^s\}$, щоб натренувати модель для $P(Y^t|X^t)$. Цей процес отримав назву трансферне навчання (transfer learning).

При цьому розподіляють гомогенне трансферне навчання, коли вихідні та цільові дані представлені у одному просторі ознак, тобто $X^t = X^s$ і $P(X^t) \neq P(X^s)$ та гетерогенні трансферне навчання, при якому вихідні та цільові дані представлені по різному, тобто $X^t \neq X^s$. Різниця між даними може бути навіть більш істотною. Так наприклад, ми можемо мати справу з зображеннями у вихідних даних та текстом у цільових.

Виходячи з усього вище вказаного, усі підходи до трансферного навчання можна поділити на три основні групи в залежності від того, як між собою відрізняються вихідний та цільовий домени та яка задача ставиться у вихідному та цільовому доменах. Виділяють індуктивне, трансдуктивне та спонтанне трансферне навчання.

Індуктивне трансферне навчання розглядає випадок, коли цільова та вихідна задачі відрізняються, але дещо схожі. При цьому не має значення чи співпадають цільовий та вихідний домени, але вихідний датасет має містити розмічені дані для побудови моделі прогнозувань.

При трансдуктивному навчанні вихідна та цільова задача однакова, але дані у цільовому та вихідному датасетах представлені по різному, тобто $X^t \neq X^s$.

При спонтанному трансферному навчанні розглядається випадок, коли і цільовий і вихідний домени різні, але мають якусь залежність. При цьому мітки відсутні як в цільовому так і в вихідному доменах. До цієї групи відносять проблеми кластерізації, зменшення розмірності та оцінки щільності[6].

За цією класифікацією, доменну адаптацію слід віднести до методів трансдуктивного трансферного навчання, оскільки задачі поставлені у доменах однакові, тобто $\tau^t = \tau^s$. Загальний випадок виникає в таких ситуаціях, коли обидва домени мають однаковий набір міток та розподіли умовних вірогідностей, тобто $y^t \neq y^s$ та $P(Y|X^t) = P(Y|X^s)$. Однак, при вирішенні реальних задач, друга умова не завжди виконується, тож завдання, до яких доцільно застосовувати доменну адаптацію прийнято вважати ті, при яких виконується хоча б $y^t \neq y^s$. Далі доменну адаптацію можна поділити на спонтанну (unsupervised) та напівавтоматичну (semi-supervised). При спонтанній доменній адаптації мітки присутні тільки у вихідному домені, а у напівавтоматичній – тільки невелика кількість даних з цільового датасету розмічена.

1.4 Формулювання задачі семантичної сегментації

Семантична сегментація зображення – задача розділення зображення на декілька сегментів, які відносяться до певного класу і формують об’єкти на зображенні. Задачу семантичної сегментації можна розглядати як задачу класифікації кожного пікселя на зображенні (рисунок 1.2).

Рисунок 1.2 – Семантична сегментація як попіксельна класифікація

Семантична сегментація на відміну від виявлення об’єктів не намагається побудувати рамку навколо об’єкту. Також різні об’єкти, що належать до одного класу, не відрізняються один від одного. Наприклад, усі пікселі автомобілів, що присутні на зображенні, отримають одну й ту саму мітку.

Семантична сегментація зображення має багато залузей застосування, головні з яких це сегментація зображень доріг для задач автономного транспорту[7], сегментація органів та пухлин на медичних знімках[8,9,10], сегментація геозображень[11].

Незважаючи на високі результати, які демонструють алгоритми семантичної сегментації на даний момент, існує проблема адаптації існуючих моделей для нових

доменів. Наприклад, модель для сегментації зображень сцен доріг, що натренована на зображеннях одної місцевості, часто демонструє гірші результати на зображеннях іншого міста, в особливості коли між вихідним та цільовим доменами присутній коваріантний зсув. Встановлено, що при зміні міста, в якому зроблені зображення, ефективність моделі може знизитися на 25-30% [12].

Зауважимо, що опція навчання нової моделі в даному випадку не розлядається, оскільки ми не маємо розмічених даних з нового домену, а створення анотацій вручну займає дуже багато часу. Підраховано, що розмітка пікселей в одному зображенні датасета Cityscapes в середньому займає 90 хвилин[13]. Було запропоновано декілька варінтів, яким чином автоматизувати процес створення анотацій. Серед них використання трьохмірної інформації[14], використання синтетичних даних, використання слабко контрольованого навчання для генерації міток[15]. Втім усі із зазначених підходів все ще потребують ручної анотації, тобто при великому обсязі сирцевих нерозмічених даних процес створення анотацій все одно потребує багато часу.

Варіант, який потребує значно менше часу – застосування методів адаптації домену. В результаті модель натренована на вихідному домені, буде налаштована таким чином, щоб демонструвати високу ефективінсть при роботі із зображеннями цільового домену.

1.5 Доменна адаптація засобами глибинного навчання

Зі зростанням можливостей глибинного навчання, точність моделей семантичної сегментації значно звищилася. Загальний підхід до адаптації моделі глибинного навчання до іншого домену полягає у видаленні останніх шарів нейронної мережі і повторне навчання моделі на даних з цільового домену. Оскільки перші шари

нейронної мережі відповідальні за більш загальні ознаки і дані обох доменів мають ці ознаки, ці шари можна не перенавчати. Розуміння цього процесу важливо, оскільки він лежить в основі більшості більш складних підходів доменної адаптації. Далі цей підхід розглянуто в контексті класифікації зображень, а саме для адаптації моделі VGG16 для класифікації зображень ImageNet для датасету CIFAR10[16].

Модель VGG була обрана для дослідження в даній роботі за декількох причин. По-перше, існує багато попередньо навчених моделей заснованій на цій архітектурі, що вирішують різні проблеми комп'ютерного зору. Це робить VGG дуже зручним вибором для досліджень доменної адаптації. По-друге, VGG демонструє високі результати при вирішені проблем. По-третє, ця архітектура добре підходить для вивчення та дослідження, оскільки її порівняно легко реалізувати.

Недоліками цієї архітектури є повільний темп навчання та великий розмір натренованої моделі, втім оскільки методи доменної адаптації дозволяють значно зменшити час навчання, для даної роботи повільне навчання не становить великої перешкоди.

VGG16 - модель конволюційної нейронної мережі, розроблена в рамках змагання ILSVRC-2014 і запропонована в статті "Very Deep Convolutional Networks for Large-Scale Image Recognition"[17]. Модель демонструє точність 92.7% при тестуванні на ImageNet в задачі розпізнавання об'єктів на зображенні. Датасета ImageNet складається з більш ніж 14 мільйонів зображень, кількість класів складає 1000. Вона є покращеною версією AlexNet, в якій замінені великі фільтри (розміру 11 і 5 в першому і другому конволюційному шарі, відповідно) на кілька фільтрів розміру 3x3, що йдуть один за одним.

На вхід шару conv1 подаються RGB зображення розміру 224 на 224. Далі зображення проходять через стек конволюційних шарів, в яких використовуються фільтри з дуже маленьким рецептивних полем розміру три на три (який є найменшим розміром для отримання уявлення про те, де знаходиться право, ліво, верх, низ, центр). Конволюційний крок дорівнює одному пікселю. Просторове доповнення (padding)

входу конволюційного шару вибирається таким чином, щоб просторова роздільна здатність зберігалася після згортки, тобто доповнення дорівнює одиниці для конволюційних шарів розміром три на три. Просторове об’єднання здійснюється за допомогою п'яти max-pooling шарів, які слідують за одним із конволюційних шарів. Операція max-pooling виконується на вікні розміру два на два пікселів з кроком два. Архітектура мережі зображена на рисунку 1.3.

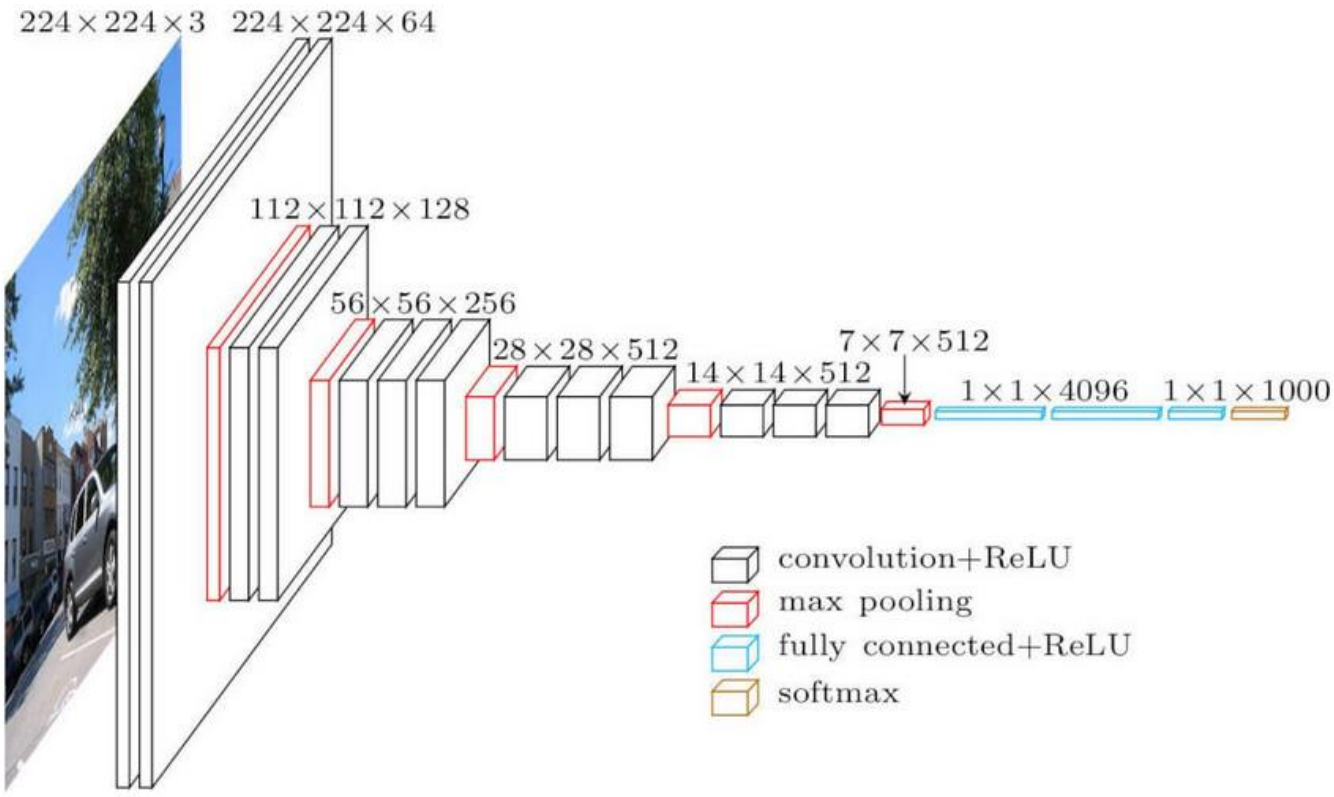

Рисунок 1.3 – Архітектура VGG16

На вхід шару conv1 подаються RGB зображення розміру 224 на 224. Далі зображення проходять через стек конволюційних шарів, в яких використовуються фільтри з дуже маленьким рецептивних полем розміру три на три (який є найменшим розміром для отримання уявлення про те, де знаходиться право, ліво, верх, низ, центр). Конволюційний крок дорівнює одному пікселю. Просторове доповнення (padding) входу конволюційного шару вибирається таким чином, щоб просторова роздільна

здатність зберігалася після згортки, тобто доповнення дорівнює одиниці для конволюційних шарів розміром три на три. Просторове об’єднання здійснюється за допомогою п'яти max-pooling шарів, які слідують за одним із конволюційних шарів. Операція max-pooling виконується на вікні розміру два на два пікселів з кроком два.

Після стеку конволюційних шарів (який має різну глибину в різних архітектурах) йдуть три повністю зв’язаних шарів: перші два мають по 4096 каналів, третій - 1000 каналів (по одному каналу на кожний з 1000 класів). Останнім йде soft-max шар. Зауважимо, що конфігурація повністю зв’язаних шарів залишається постійною у всіх нейросетях.

Всі приховані шари забезпечені ReLU. Відзначимо також, що мережі не містять шару нормалізації (Local Response Normalisation), оскільки нормалізація не покращує результату на датасеті ILSVRC, а веде до збільшення споживання пам'яті та часу виконання коду.

Зазвичай процес класифікації для мережі VGG-16 починається з введення зображення в мережу. Зображення розповсюджується по мережі коли нарешті отримується остаточна класифікація ймовірностей в кінці мережі через soft max.

Щоб налаштувати мережу в рамках трансферного навчання необхідно внести кілька змін у мережу. По-перше, необхідно заморозити частину мережі з конволюційними шарами. Це пояснюється тим, що ці шари мережі навчені вихоплювати більш абстрактні ознаки, такі як краї об’єктів, тобто ці шари повинні добре працювати на цільовому домені.

Потім необхідно або видалити повноз’єднані шари наприкінці мережі (шари відповідальні за класифікації по отриманим раніше ознакам). В нашому у нового класифікатора замість 1000 класів оригінального VGG16 для датасета ImageNet встановлюється 10 класів для CIFAR10. Після цього навчання мережі починається знову, але тільки для повноз’єднаних шарів.

Для того щоб підвищити точність класифікації мережі на цільовому датасеті необхідно можна включити до процесу навчання моделі один або декілька

конволюційних шарів наприкінці моделі. Це очікувано збільшить час на навчання моделі, але забезпечить кращі результати на цільовому домені.

Знаходження найкращого темпу навчання може сильно вплинути на час навчання моделі. Низький темп навчання призводить до довшого тренування моделі. Але при занадто високому темпу навчання існує ризик ніколи не знайти параметри, що дадуть задовільну точність.

Щоб подалати цю проблему, можна на початку епохи обирати дуже мале значення для темпу навчання і підвищувати це значення з кожною новою партією даних. Таким чином наприкінці епохи темп навчання навпаки буде дуже високим. Далі обирається той темп навчання, який забезпечує найбільший спад. Графік темпів навчання, отриманих для даної задачі наведено на рисунку 1.4.

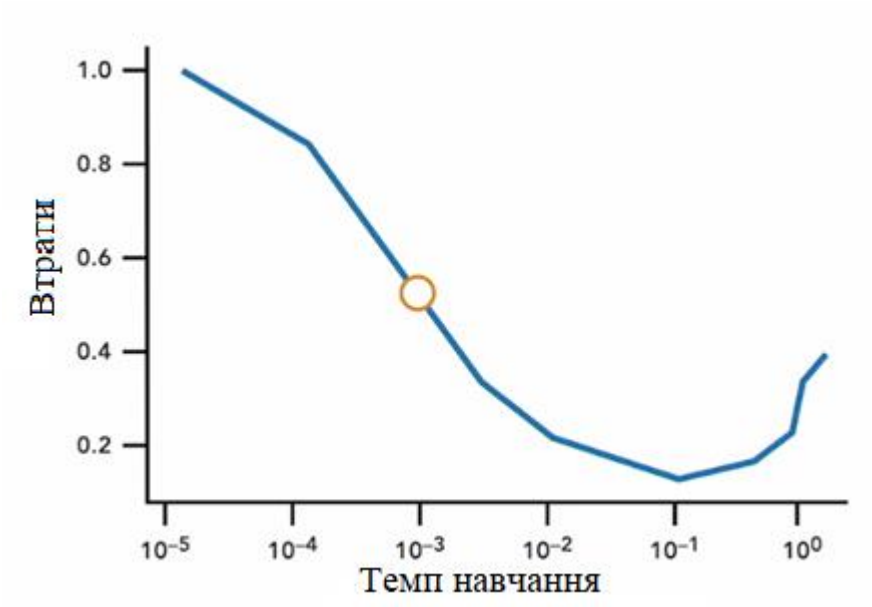

Рисунок 1.4 – Темпи навчання отримані для алгоритму оптимізації Adam на датасеті CIFAR10

Виходячи з отриманих результатів робимо висновок, що оптимальними значеннями для темпу навчання є $e \cdot 10^{-4}$ або $e \cdot 10^{-3}$.

1.6 Методи адаптації домену з використанням псевдоміток

Описаний підхід до доменної адаптації не працює у ситуації, коли вихідний домен має дані з мітками, а цільовий – ні. Така ситуація дуже поширена в задачах семантичної сегментації.

Рішення, яке найчастіше застосовується в такій ситуації, – натренувати модель на вихідному датасеті, подати певну кількість зображень з цільового датасета до натренової моделі, додати отримані розмічені зображення до вихідного датасету, щоб повторно натренувати модель. Така послідовність дій повторюється *n* раз, доки модель не продемонструє задовільного результату на цільовому датасеті. Мітки, отримані таким чином на цільовому датасеті, носять назву псевдоміток.

При такому підході слід вирішити, які зображення відбирати на кожній ітерації. Можна використати логістичну регресію, щоб обримати вірогіднсть того, наскільки оброблене зображення розмічено правильно. Так, якщо вірогідність перевищує певне граничне значення, псевдомітки додаються до наступної ітерації. З кожною наступною ітерацією граничне значення зменшується.

В адаптації домену методом опорних векторів поступово обирається 2 тисячі зразків з вихідного датасета, які замінюються на таку ж кількість зразків цільового датасета. Зразки обрані з цільового датасета попередньо розмічуються. Метод застосовує дві функції витрат: для вихідного датасета та для цільового датасета. Поступово збільшуючи коефіцієнт вартості для цільового датасета та зменшуючи коефіцієнт вартості для вихідного датасета, модель адаптується до цільового домену.

Якщо розглядати цільові мітки як приховані змінні, тоді тренування можна провести за допомогою процедури максимізації очікування. Доменна адаптація за цим методом використовує модель для отримання міток на зразках цільового датасета.

Перевагою методів заснованих на генерації псевдоміток є можливість використовувати псевдомітки отримані зі зразків цільового домену у якості

валидаційного датасету, втім розглянуті методи генерації псевдоміток мають і спільний недолік.

Розглянемо ситуацію, коли в цільовому та вихідному доменах існують класи, які сильно відрізняються між доменами та класи, які дуже схожі, наприклад, при адаптації моделі, натренованої на домені дорожніх зображень мегаполіса до домену дорожніх зображень невеличкого міста. Оскільки об’єкти деяких класів схожі в обох доменах (світлофори, транспорт), модель буде обирати з цільового домену зразки з цими об’єктами для створення псевдоміток. Зразки, які мають об’єкти класів, що відрізняються в доменах будуть ігноруватися (тротуари, домівки вздовж доріг, паркани). В результаті модель продемонструє високі показники точності при сегментації світлофорів та транпорту і буде цілком нездатна сегментувати тротуари, домівки, тобто отримана модель буде схилятися до певних класів, ігноруючи інші.

1.7 Постановка задачі

Аналіз методів з генерацією псевдоміток для адаптації домена в галузі семантичної сегментації продемонстрував необхідність вирішення проблеми балансування класів. Завдання, яке ставить перед собою дана робота – дослідити, як можна подолати проблему, коли модель, адаптована за методом генерації псевдоміток, схиляється до одних класів та ігнорує інші. Для цього дослідити, яким чином можна виконати балансування класів в інших задачах машинного навчання, запропонувати новий метод або покращення існуючого, обрати датасети та модель для експерименту, дослідити запропонований підхід балансування класів при адаптації моделі, порівняти результати роботи модифікованої моделі з оригінальною моделлю та моделлю натренованою на цільовому датасеті. На основі цих порівнянь зробити висновок щодо ефективності методу, подальших досліджень.

2 ОГЛЯД МЕТОДІВ ДОМЕННОЇ АДАПТАЦІЇ В КОНТЕКСТІ СЕМАНТИЧНОЇ СЕГМЕНТАЦІЇ

2.1 Методи машинного навчання для адаптації домену

Перш ніж впроваджувати нові методи доменної адаптації для проблем комп'ютерного зору, необхідно провести аналіз існуючих методів. Оскільки багато методів, заснованих на глибинному навчанні використовують простіші класичні алгоритми, доцільним буде розглянути ці прості алгоритми в першу чергу, потім перейти до глибинного навчання. Далі приведено перелік методів доменної адаптації.

Випадок, коли розподіли умовних вірогідностей однакові в обидвах доменах, тобто $P(Y|X^s) = P(Y|X^t)$ отримав назву коваріантний зсув. Тоді можна просто застосувати модель, отриману при тренуванні на вихідному датасеті для цільових даних $P(Y|X^t)$. Однак, оскільки $P(X^s) \neq P(X^t)$, нема гарантій, що модель продемонструє високу точність роботи з даними цільового домену. Одним з рішень поліпшення моделі є перерахунок вагових коефіцієнтів[18].

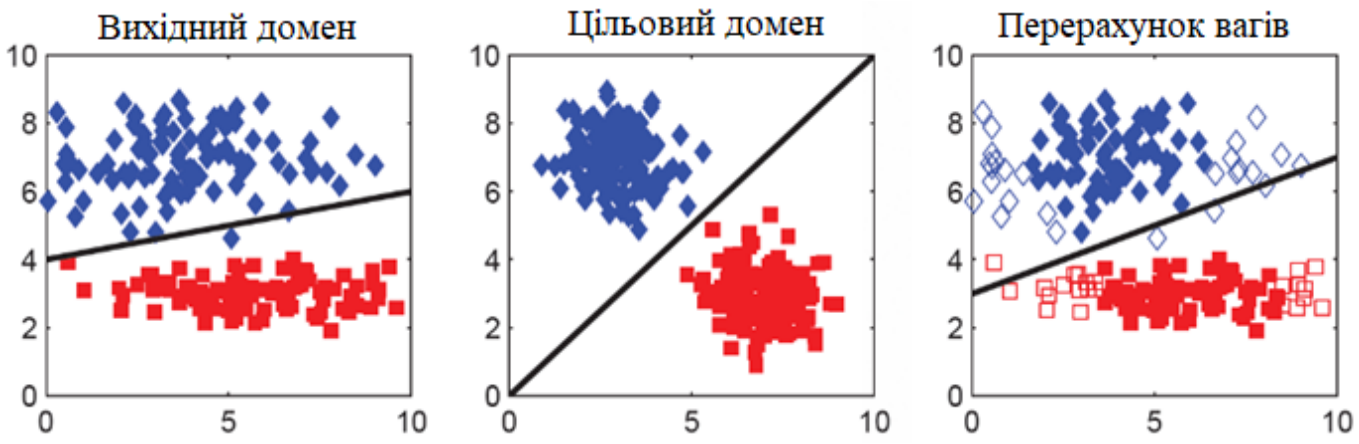

Рисунок 2.1 – Ілюстрація перерахунку елементів вихідного класифікатора

В основі метода полягає ідея ввести коефіцієнти для вихідного датасета щоб модифікувати дані. Внесені зміни повинні призвести до того, щоб натренована на зкорегованих даних модель добре працювала з даними із цільового датасета (рис. 2.1).

Таким чином постає питання як саме провести розрахунок вагових коефіцієнтів. Підрахувати вагу елементу датасета можна як відношення вірогідностей належності елементу вихідному або цільовому домену.

Метод перерахунку вагових коефіцієнтів реалізовано в трансферному адаптивному бустінгу (TrAdaBoost)[19], який ітеративно перераховує вагові коефіцієнти даних як у вихідному так і цільовому датасетах під час навчання класифікатора для цільового датасету. Разом з цим проводиться зменшення вагових коефіцієнтів неправильно класифікованих даних вихідного датасету з метою зменшення їхнього впливу на процес навчання.

Наступна група отримала назву методів адаптації параметра. Методи цієї групи змінюють параметри моделі таким чином, щоб класифікатор, навчений на вихідному домені, демонстрував високі результати класифікації на цільовому домені. Таким чином, робота таких методів полягає у зменшенні помилки узагальнення шляхом введення до процесу навчання моделі інформації про цільові дані. Прикладом таких методів є Adaptive SVM, який поступово корегує рішення вихідного класифікатора завдяки функціям ускладнення (perturbation functions), так що класифікатор адаптується до даних цільового домена.

Метод нарощення ознак (feature augmentation)[20] для полягає в тому, що дані X доповнюються собою та вектором такого ж розміру заповненого нулями. Вихідні ознаки стають $\begin{bmatrix} x^s \\ x^s \\ 0 \end{bmatrix}$, а цільові ознаки стають $\begin{bmatrix} x^t \\ 0 \\ x^t \end{bmatrix}$. Потім метод опорних векторів тренується на доповнених ознаках, щоб знайти, які дані відносяться до обох доменів і які належать одному домену.

Ідея розширення ознак лягла в основу таких методів як Geodesic Flow Sampling, Geodesic Flow Kernel, Reproducing Kernel Hilbert Space.

В методі вирівняння простору ознак[21] вихідний та цільовий домени представляються як підпростори створені з власних векторів. Метод намагається

провести доменну адаптацію шляхом визначення функції, що переводить вихідний підпростір у цільовий. До методів цієї групи відносять Subspace Alignment, Correlation Alignment та інші.

Ще одна група методів – перетворення ознак на основі вивчення метрік. Методи цієї групи відносяться до контрольованого перетворення ознак і припускають, що хоча б невелика кількість міток з цільового домена доступна. Для того щоб перейти від вихідного домену до цільового використовуються метричні методи навчання. Прикладом такого підходу є доменна адаптація на базі класифікатора Naive Bayes Nearest Neighbor.

Хоча вище вказані методи є найбільш відомими, неглибинні методи доменної адаптації не обмежуються лише ними, але розгляд цих методів виходить за рамки даної роботи.

2.2 Методи глибинного навчання для адаптації домену

Усі моделі, що вирішують задачі адаптації домену за допомогою глибинного навчання, можна поділити на три групи. Методи першої групи використовують конволюційні моделі для виділення векторних ознак і далі аналізують ці ознаки за допомогою неглибинних методів доменної адаптації. Методи другої групи навчають або налаштовують модель на вихідному домені, редагують її для задачі цільового домену і застосовують отриману модель на цільовому домені. Методи третьої групи вводять нові архітектури нейронних мереж (наприклад генеративно-змагальні мережі), що розроблені спеціально для проблем доменної адаптації.

Методи першої групи застосовують нейронну мережу виключно як засіб виділення ознак. Ознаки, що отримані таким чином з вихілного та цільового доменів можуть далі оброблятися одним з методів неглибинного навчання, розглянутих вище.

Загалом такий підхід не призводить до суттєвих поліпшень методів неглибинного навчання і на даний момент застосовується нечасто.

Другий підхід полягає у налаштуванні вже натренованої моделі на нових даних або для нової задачі[22]. Зауважимо, що таке налаштування потребує великих обсягів розмічених даних, які часто відсутні у цільовому домені або присутні в невеликому обсязі. Тоді налаштування моделі проводиться на вихідному домені, що розширюється за рахунок розмічених даних цільового домену. За відсутності таких розмічених даних, вдаються до генерації псевдоміток. За рахунок цього модель редагується для нової задачі. Також важливо зазначити, що при такому підході важливо відбирати дані, які погано представлені у вихідному датасеті. В іншому разі налаштування моделі на вихідному домені може призвести до надмірного навчання на вихідному датасеті. Це в свою чергу призведе до поганих показників роботи моделі на цільовому домені.

Прикладом архітектури нейронної мережі, що відноситься до групи змагально-диференціальних моделей, є доменно-змагальні нейронна мережа[23] (Domain-Adversarial Neural Network). Архітектура мережі наведена на рисунку 2.2.

Архітектура доменно-змагальної нейронної мережі містить функцію витягнення ознак (зображено зеленим кольором) та класифікатором (зображено синім кольором), які разом утворюють стандартну архітектуру прямого поширення. Доменна адаптація досягається за рахунок введення градієнтного зворотнього шару, що помножує градієнт на певну негативну константу під час зворотнього поширення. Завдяки цьому розподіли ознак у двох доменах стають нерозрізненими.

Моделі групи змагально-генеративних моделей додають до диференціальної моделі генеративний компонент, що зазвичай представляє собою генеративно-змагальну мережу. Прикладом такої архітектури є з’єднані генеративно-змагальні мережі (coupled generative adversarial network)[24]. Ця модель містить ряд генеративно-змагальних мереж, кожна з яких відноситься до одного з доменів. Модель навчають на спільному розподілі зображень з доменів.

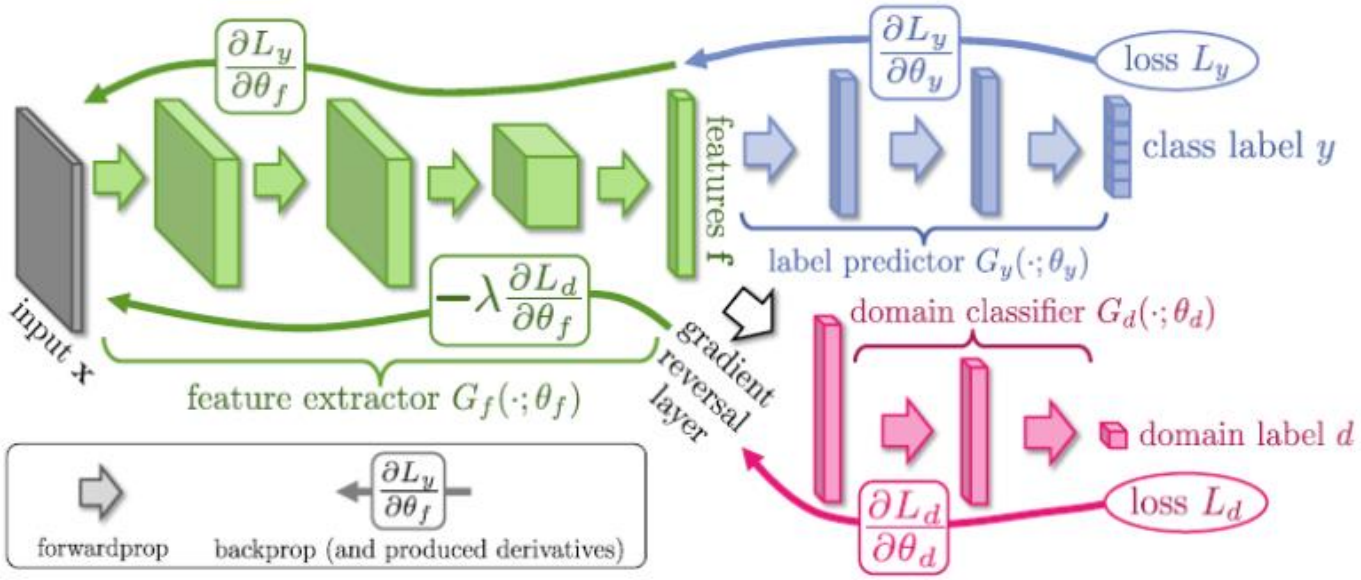

Рисунок 2.2 – Архітектура доменно-змагальної нейронної мережі

Інший підхід, що характерний для методів цієї групи полягає у розширенні вихідного домену шляхом генерації зображень на основі цільового домену, при цьому згенеровані зображення виглядають як зображення з цільового домену.

Методи засновані на реконструкції даних поєднують стандартну конволюційну нейронну мережу для вихідного класифікатора зі зворотньою конволюційною мережею для реконструкції даних цільового домену. Щоб навчити модель як класифікації на вихідному домені так і реконструкції даних вихідного домену, слід поперемінно робити неконтрольоване та контрольоване навчання.

2.3 Огляд датасетів для семантичної сегментації

Датасет PASCAL VOC[25] містить 11 тисяч зображень, що відводяться для навчання та валідації, і 10 тисяч зображень, що формують датасет для тестування. У якості метрики оцінювання обрано середнє відношення перетину до сукупності

(mIoU). Метрика mIoU підраховує середнє відношення області перетину та сукупності областей для отриманої та реальної областей для всіх зображень.

Датасет PASCAL-Context на відміну від PASCAL VOC складається з повністю розмічених зображень, тобто зображення цільком поділена на сегменти за 400 категоріями. Датасет містить по 10 тисяч зображень для тренування, валідації та тестування. Метрика – mIoU.

Датасет COCO для сегментації містить 200 тисяч зображень, загальна кількість сегментованих об'єктів перевищує 500 тисяч[26]. Датасет поділений на тренувальну, валідаційну частини та частину для тестування. Частина для тестування містить об'єкти з 80 категорій і тільки ці об'єкти розмічені на зображеннях з цієї частини. Застосовуються наступні метрики: середня точність та середня повнота, які вираховуються через відношення перетину до сукупності.

Датасет Cityscapes містить розмічені зображення урбаністичних сцен з 50 міст. Датасет відводить 23.5 тисячі зображень для тренування та валідації та приблизно півтори тисячи зображень для тестування. Кожне зображення датасета повністю поділене на сегменти за 29 категоріями. Датасет дуже часто використовується для вирішення задач семантичної сегментації, пов'язаних з самокерованими автомобілями.

Датасет SYNTHIA складається з синтетичних зображень, розроблений спеціально для семантичної сегментації в контексті вирішення задач управління транспортом. Зображення розмічені на сегменти за 13 класами. Загальна кількість зображень перевищує 200 тисяч. Крім безпосередньо зображень датасет містить дані з 8 симульованих сенсорів.

Датасет GTA містить 24966 розмічених кадрів. Цей датасет часто використовують разом з датасетами CamVid та CityScapes, щоб розширити вихідний домен.

Наведений перелік датасетів для семантичної сегментації не є повним. Втім зазначимо, що в більшості випадків при дослідженні та проектуванні рішень

семантичної сегментації застосовуються саме ці датасети. Окрім цього, в зв'язку з їх поширеністю дослідження семантичної сегментації на одному з цих датасетів дозволить порівняти отримані результати з сучасними методами.

2.4 Моделі глибинного навчання для задач семантичної сегментації

Найпростіший підхід до вирішення задачі семантичної сегментації засобами глибинного навчання полягає у використанні звичайних конволюційних нейронних мереж. Розгляд цієї моделі важливий, оскільки вона складає базу, на основі якої в подальшому буде побудовано багато рішень для семантичної сегментації. Результати оцінюються за метрикою mIoU.

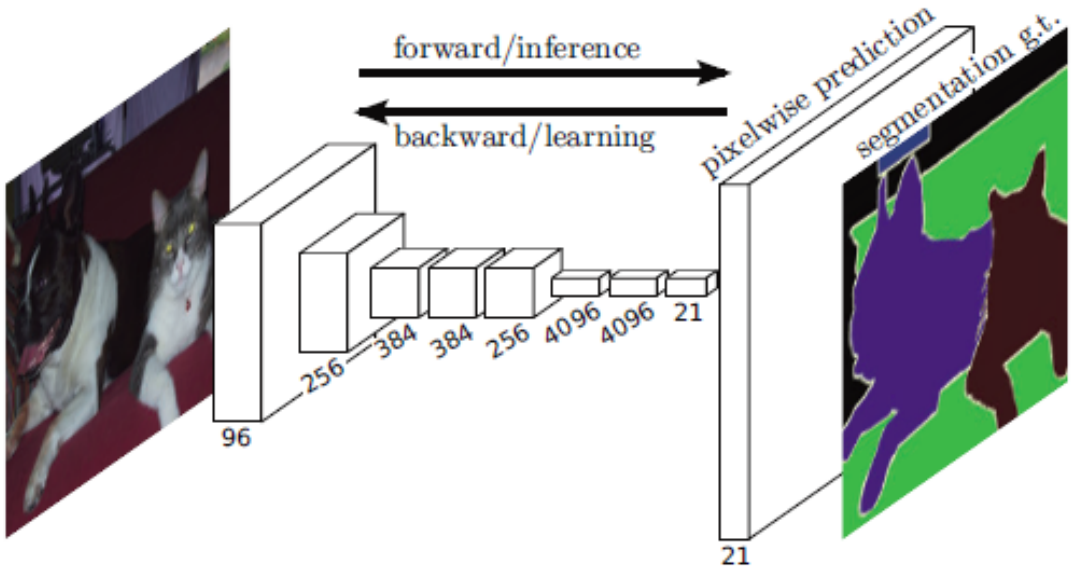

Рисунок 2.3 – Архітектура повністю конволюційних мереж

Щоб адаптувати конволюційні мережи до задачі сегментації, необхідно замінити усі повнозв'язні шари в кінці мережи на конволюційні. Тоді на виході такої мережи отримуємо певну кількість карт ознак малого розміру. Щоб отримати

сегментоване зображення такого ж розміру як і вхідне, необхідно додати деконволюційні шари до архітектури мережі. Архітектура мережі умовно наведена на рисунку 1. На вхід до повністю конволюційної мережи подається зображення довільного розміру, результатом роботи мережи є розмічене зображення. Розмір вихідного зображення дорівнює вхідному.

При вирішенні задачі семантичної сегментації на датасеті 2012 PASCAL VOC за метрикою mIoU було отримано результат у 62.5%.

Наступною архітктурою, яка з'явилася після повністю конволюційних мереж є ParseNet[27]. Автори ParseNet стверджують, що головний недолік повністю конволюційних мереж полягає у втраті глобального контексту зображення при збільшенні шарів. Схема роботи ParseNet наведена на рисунку 2.4.

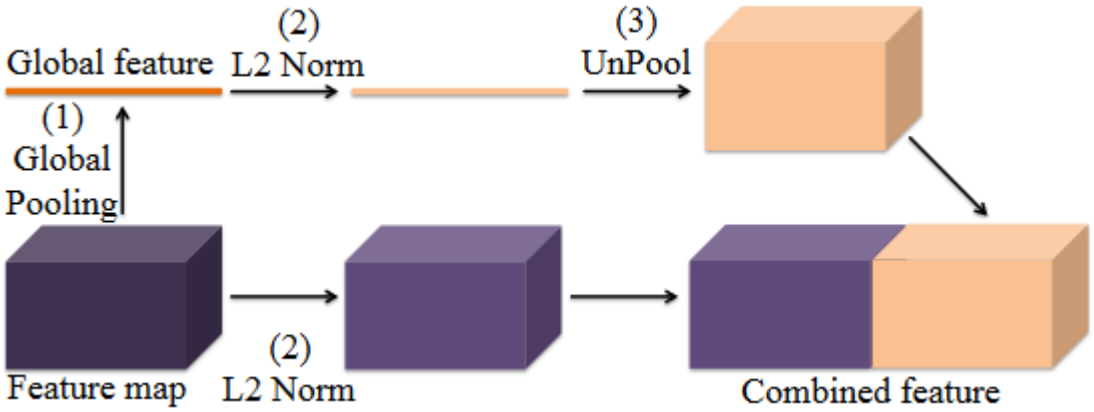

Рисунок 2.4 – Схема роботи ParseNet

Спочатку модель створює карти ознак, які перетворюються у єдиний вектор глобальних ознак, за яким слідує операція агрегації за максимумом. Потім проводиться нормалізація усіх початкових карт ознак і об'єднання з карт ознак, отриманих раніше. Таким чином, ParseNet представляє собою модифіковану повністю конволюційну мережу, де конволюційні шари були заміщені на описану вище структуру. ParseNet продемонструвала кращий результат ніж чисті повністю

конволюційні мережі, отримавши оцінку 68.9% за метрикою mIoU на датасеті 2012 PASCAL VOC.

U-Net[28] – ще одна модифікація повністю конволюційних мереж. U-Net скаладається з двох частин. Перша частина відповідальна за виявлення ознак із зображення, друга – за розширення карт ознак для сегментації всього зображення. Зазначена архітектура наведена на рисунку 2.5.

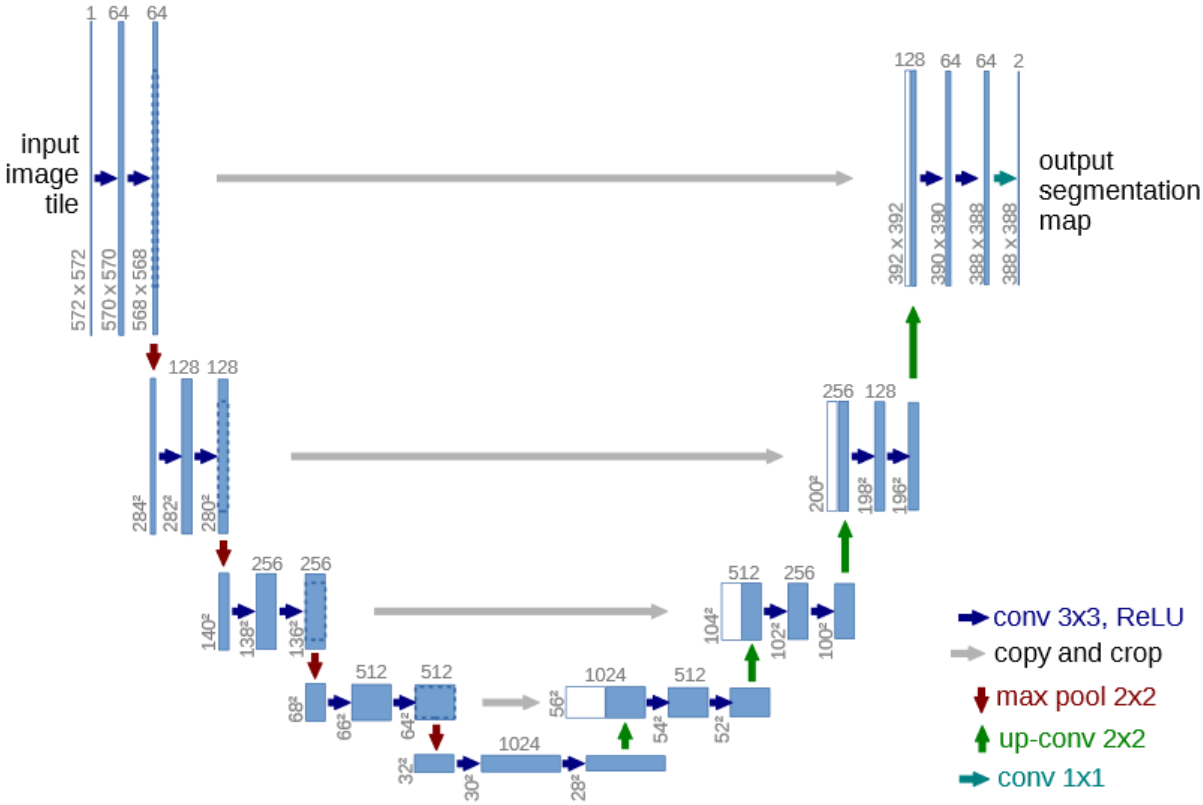

Рисунок 2.5 – Архітектура U-Net

Згорткова частина складається з повністю конволюційної мережи, що використовує фільтр розміром 3x3. Після кожного конволюційного шару йде випрямляч ReLU. Після двох конволюційних шарів слідує операція агрегації за максимом з кроком 2. При кожній операції зменшення кількість карт ознак збільшується вдвічі.

Деконволюційна частина поступово розповсюджує отримані ознаки на усе зображення. Це досягається шляхом операції деконволюції, яка збільшує розмір карт

ознак зменшуючи їхню кількість вдвічі. За цією операцією слідують два конволюційних шари.

U-Net продемонструвала високі показники при вирішені задач семантичної сегментації для медичної галузі, але в інших доменах (автономний транспорт) показали значно гірші результати. Також важливою характеристикою цієї архітектури є можливість навчання з малих датасетів і помітно менший час тренування.

Архітектура Mask R-CNN[29] базується на Faster R-CNN, яка використовується для детекції об’єктів. Faster R-CNN складається з двох компонент. Одна відповідальна за витягування ділянок зображення, які представляють інтерес. Інша стягує ознаки з цих ділянок і виводить клас та координати класифікованого об’єкту.

Mask R-CNN додає третю гілку виводу для Faster R-CNN, яка містить бінарну маску, за якою проводиться сегментація об’єкта у рамці. Мережа Mask R-CNN показала гарний результат на датасеті COCO досягнувши оцінки 40.9% у задачі семантичної сегментації за метрикою mIoU.

Архітектура Feature Pyradmid Network складається з проходення знизу вгору, зверху вниз і модулю з’єднання, що поєднує низко-полігональні ознаки та високо-полігональні ознаки. На вхід до проходження знизу вгору подається зображення довільного розміру, яке обробляється конволюційними шарами і зменшується шарами агрегації за максимумом. Звернемо увагу на те, що кожна група карт ознак одного розміру називається етапом. Вихідні дані на кожному етапі утворюють рівень піраміди.

При проходження згори донизу виконується розширення останніх карт ознак і об’єднання цих ознак з ознаками того ж етапу отриманих при проходженні знизу вгору. Карти ознак отримані при проходженні знизу вгору обробляються конволюційним фільтром розміром 1x1 і зливаються з картами ознак отриманими при проходженні зверху вниз. Отримані карти ознак оброблюються конволюційним фільтром 3x3 і на кожному етапі проходження згори вниз створюється прогноз для детекції об’єкта. Для семантичної сегментації зображень автори мережі пропонують

використовувати два багатошарових перцептрона для створення двох масок різного розміру для об’єктів. Зазначена архітектура зображена на рисунку 2.6.

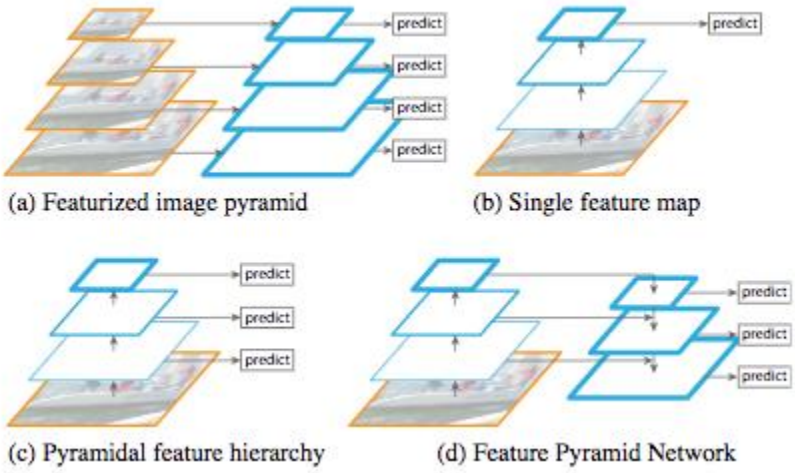

Рисунок 2.6 – Архітектура FPN

Отримані карти ознак оброблюються конволюційним фільтром 3x3 і на кожному етапі проходження згори вниз створюється прогноз для детекції об’єкта. Для семантичної сегментації зображень автори мережі пропонують використовувати два багатошарових перцептрона для створення двох масок різного розміру для об’єктів. Запропонований метод дуже ефективний, оскільки при ньому краще поширюються низкополігональні дані через мережу. В 2016 році модель за цією архітектурою досягла результату у 48.2% на датасеті COCO за метрикою середньої повноти.

3 ДОСЛІДЖЕНННЯ МЕТОДУ ДОМЕННОЇ АДАПТАЦІЇ З БАЛАНСУВАННЯМ КЛАСІВ

3.1 Доменна адаптація для семантичної сегментації без балансування класів

Припустимо, що вирішується задача, в якій мітки доступні як у вихідному, так і у цільовому доменах. Контрольованим налаштуванням називатимемо процес перенавчання останніх шарів нейронної мережі з метою зміни вагових коефіцієнтів і відповідно покращення моделі для визначеного домену.

Тоді можна провести доменну адаптацію шляхом контрольованого налаштування моделі для обох доменів даних. Приймаючи до уваги, що останній шар нейронної мережи представляє собою результат багатозмінної логістичної функції (softmax) проблему адаптації домену можна сформувати як функцію мінімізації наступним чином:

$$\min_{w} \zeta_s(w) = -\sum_{s=1}^{S}\sum_{n=1}^{N} y_{s,n}^{T} \log\big(p_n(w, I_s)\big) - \sum_{t=1}^{T}\sum_{n=1}^{N} y_{t,n}^{T} \log\big(p_n(w, I_t)\big) \qquad (3.1)$$

де I_s означає зображення з вихідного домену з порядковим номером $s = 1,2,\dots,S$, $y_{s,n}$ відповідає мітці з вхідного датасета для пікселя за номером n у зображенні I_s, n в свою чергу змінюється як $n = 1,2,\dots,N$, $p_n(w, I_s)$ – результат багатозмінної логістичної функції, що містить розподіл ймовірностей класів для пікселя n з зображення I_s, w – це матриця з ваговими коефіцієнтами нейронної мережі, I_t відповідно означає зображення з цільового домену з порядковим номером $t = 1,2,\dots,T$, $y_{t,n}$ відповідає мітці з цільового датасета для пікселя за номером n у зображенні I_t, $p_n(w, I_t)$ – результат багатозмінної логістичної функції, що містить розподіл ймовірностей класів для пікселя n зображення I_t.

Однак даний підхід не працює для випадків семантичної сегментації, коли в цільовому домені відсутні розмічені пікселі. Одним із шляхів подолання цієї проблеми є створення псевдоміток. Для цього дані з цільового домену подаються на вхід класифікатора, і таким чином отримуються розмічені дані з цільового домену. Це дозволяє застосовувати методи характерні для задач доменної адаптації, де розмічені дані і цільового і вихідного датасетів.

Щоб адаптувати функцію мінімізації для випадку, коли цільові дані не розмічені, можна розглядати розмічені класи пікселів з цільового датасета як приховані змінні, що поступово отримуються під час навчання моделі. В такому випадку функція приймає вигляд:

$$\min_{w,\hat{y}} \zeta_U(w,\hat{y}) = -\sum_{s=1}^{S}\sum_{n=1}^{N} y_{s,n}^{T} \log\big(p_n(w,I_s)\big) - \sum_{t=1}^{T}\sum_{n=1}^{N} \hat{y}_{t,n}^{T} \log\big(p_n(w,I_t)\big) \qquad (3.2)$$
$$\hat{y}_{t,n} \in \left\{e^{(i)} \middle| e^{(i)} \in \mathbb{R}^C\right\}, \forall t,n$$

де $\hat{y}$ – цільові мітки, C – кількість класів

Аргумент $e^{(i)}$ представляє собою вектор, усі елементи якого дорівнюють нулю, окрім i-го елементу, що дорівнює 1. В даній функції $\hat{y}$ можна вважати псевдомітками.

Головна проблема, яка постає при генерації псевдоміток – це їхня коректність. Зрозуміло, що чим більша різниця між цільовим та вихідним доменами, тим більша вірогідність, що отримані псевдомітки некоректні.

Для вирішення цього запропоновано наступний підхід: розумніше обирати ті дані з цільового датасету, в яких ймовірно найбільша кількість коректно класифікованих пікселів. Ці дані включаються до процесу навчання, таким чином через декілька епох навчання отримуємо модель, що більш адаптована до цільового домену. Далі повторюємо ітерацію для зображень цільового домену, що залишилися. Зменшуємо поріг ймовірності правильно класифікованих пікселів і обираємо наступні

зображення, робимо налаштування моделі. Таким чином з кожною ітерацією кількість нерозмічених даних буде зменшуватися, а модель поступово адаптується до цільового домену. Для описаного підходу функція мінімізації приймає наступний вигляд:

$$\begin{aligned}\min_{w,\hat{y}} \zeta_{ST}(w,\hat{y}) = &-\sum_{s=1}^{S}\sum_{n=1}^{N} y_{s,n}^{T}\log\big(p_n(w,I_s)\big) \\ &-\sum_{t=1}^{T}\sum_{n=1}^{N}[\hat{y}_{t,n}^{T}\log\big(p_n(w,I_t)\big) + k|\hat{y}_{t,n}|_1] \\ &\hat{y}_{t,n} \in \left\{e^{(i)}\middle| e^{(i)} \in \mathbb{R}^C\right\} \cup 0, \forall t,n;\ k>0\end{aligned} \tag{3.3}$$

В цій функції параметр k вказує кількість псевдоміток, що ігнорується. При збільшенні k, збільшується кількість псевдоміток, що використовується при тренуванні моделі. Також звернемо увагу, що якщо для пікселя $y_{s,n}$ дано значення нуль, то відповідна псевдомітка ігнорується і не використовується при навчанні.

Для виконання мінімізації виконується наступна послідовність дій:

- редагування значень вагових коефіцієнтів з подальшим розрахунком функції мінімізації від параметра $\hat{y}_{t,n}$;
- редагування $\hat{y}_{t,n}$ та виконання класифікації пікселів в залежності від вагових коефіцієнтів.

Відповідно процес адаптації домену для семантичної сегментації складається з певної кількості ітерацій описаних двух дій. Спочатку з цільового домена обирається декілька кількість псевдоміток, потім обрані псевдомітки використовуються при навчанні нейронної мережі при виконанні алгоритму зворотнього розпоширення помилки. В результаті отримуємо модель, що адаптована до цільового домену.

Процес навчання моделі у другому пункті цілком зрозумілий, проте не зрозуміло яким чином у першому пункті необхідно обирати псевдомітки, які з

найбільшою вірогідністю коректно класифіковані. Задачу для першого пункту можна виразити наступним чином:

$$\min_{\hat{y}} -\sum_{t=1}^{T}\sum_{n=1}^{N}[\sum_{c=1}^{C}\hat{y}_{t,n}^{(C)}\log\big(p_n(c|w,I_t)\big)+k|\hat{y}_{t,n}|_1]$$
$$\hat{y}_{t,n}=[\hat{y}_{t,n}^{(1)},\dots,\hat{y}_{t,n}^{(C)}]\in\left\{e^{(i)}\middle|e^{(i)}\in\mathbb{R}^C\right\}\cup 0,\forall t,n;\ k>0 \tag{3.4}$$

Приймаючи до уваги, що $\hat{y}_{t,n}$ представляє собою вектор, елементи якого складають нулі, окрім одного елементу, який дорівнює одиниці, оптимізація псевдоміток виконується за наступною функцією:

$$\hat{y}_{t,n}^{(c)*}=\begin{cases}1, if\ c=\arg\max p_n(c|w,I_t),\\ \quad p_n(c|w,I_t)>\exp(-k)\\ 0, \qquad otherwise\end{cases} \tag{3.5}$$

На відміну від більш тривіальних методів самостійної адаптації моделі, що формують доменно-інварінтний класифікатор, методи, засновані на конволюційних нейронних мережах також здатні виокремлювати ознаки як у цільовому так і у вихідному доменах.

3.2 Доменна адаптація для семантичної сегментації з балансуванням класів

При описаному підході із застосуванням псевдоміток може виникнути ситуація, коли отримана модель добре виконує задачу семантичної сегментації для одних класів об’єктів і повністю ігнорує інші класи. Така ситуація виникає, коли під час тренування нейронної мережи були використані лише псевдомітки одних класів, а деякі класи

могли або взагалі не потрапити у процес тренування, або потрапити у останні ітерації описаного вище методу. Щоб подолати цю проблему, додано нормалізацію порогів впевненості коректності класифікації для псевдоміток. Функція мінімізації приймає наступний вигляд:

$$\begin{gathered}\min_{w,\hat{y}} \zeta_{BALANCED}(w,\hat{y}) = -\sum_{s=1}^{S}\sum_{n=1}^{N} y_{s,n}^{T}\log\big(p_n(w,I_s)\big) \\ -\sum_{t=1}^{T}\sum_{n=1}^{N}\sum_{c=1}^{C}[\hat{y}_{t,n}^{(c)}\log\big(p_n(c|w,I_t)\big) + k_c\hat{y}_{t,n}^{(c)}|_1] \\ \hat{y}_{t,n} = \left[\hat{y}_{t,n}^{(1)},\dots,\hat{y}_{t,n}^{(C)}\right] \in \left\{e^{(i)}\middle|e^{(i)} \in \mathbb{R}^C\right\} \cup 0, \forall t,n;\ k_c > 0, \forall c\end{gathered} \tag{3.6}$$

де параметр k_c вказує кількість псевдоміток класу c, що будуть приймати участь у навчанні.

Відповідно зі зміною k_c до моделі додаються різні рівні зміщення класів для псевдоміток. Це дозволяє вирішити задачу балансування класів.

При цьому оптимізація при такому підході залишається як у формулі 3.4, не зважаючи на створення псевдоміток. Загалом, оптимізацію можна виразити як

$$\begin{gathered}\min_{\hat{y}} -\sum_{t=1}^{T}\sum_{n=1}^{N}\sum_{c=1}^{C}[\hat{y}_{t,n}^{(c)}\log\big(p_n(c|w,I_t)\big) + k_c\hat{y}_{t,n}^{(c)}] \\ \hat{y}_{t,n} = \left[\hat{y}_{t,n}^{(1)},\dots,\hat{y}_{t,n}^{(C)}\right] \in \left\{e^{(i)}\middle|e^{(i)} \in \mathbb{R}^C\right\} \cup 0, \forall t,n;\ k_c > 0, \forall c\end{gathered} \tag{3.7}$$

З функції 3.7 можна вивести рівняння для псевдомітки $\hat{y}_{t,n}^{(c)*}$. Ця функція наведена у формулі 3.8.

З функції можна зробити висновок, що створення псевдоміток тепер не залежить від $p_n(c|w,I_t)$, а залежить від нормалізованого значення $\frac{p_n(c|w,I_t)}{\exp(-k_c)}$.

Створення псевдоміток через дану нормалізацію гарантує участь міток з відносно малою вірогідністю у навчанні, при чому обрані мітки матимуть найвищу вірогідність серед міток свого класу.

$$\hat{y}_{t,n}^{(c)*} = \begin{cases} 1, if\ c = \arg\max \dfrac{p_n(c|w, I_t)}{\exp(-k_c)}, \\ \qquad \dfrac{p_n(c|w, I_t)}{\exp(-k_c)} > 1 \\ 0, \qquad otherwise \end{cases} \tag{3.8}$$

З рівняння 3.8 можна зробити висновок, що створення псевдоміток тепер не залежить від $p_n(c|w, I_t)$, а залежить від нормалізованого значення $\frac{p_n(c|w, I_t)}{\exp(-k_c)}$. Створення псевдоміток через дану нормалізацію гарантує участь міток з відносно малою вірогідністю у навчанні, при чому обрані мітки матимуть найвищу вірогідність серед міток свого класу.

У випадках, коли існує декілька класів, таких що $\frac{p_n(c|w, I_t)}{\exp(-k_c)} > 1$, обирається клас з найбільшою оцінкою.

Параметр k є ключовим компонентом при відбору псевдоміток для тренування. Він залишає ті псевдомітки, вірогідність правильної класифікації яких становить більше ніж $\exp(-k)$. Параметр можна змінювати наступним чином, так щоб кількість псевдоміток у кожній ітерації збільшувалася.

Сортуємо у порядку зменшення вірогідності коректності пікселі цільового датасету. Для параметра k встановлюється так, що $\exp(-k)$ дорівнює вірогідності псевдомітка, яка знаходиться під номером $round(p * T * N)$ серед відсортованих міток, де p − число у межах [0,1] є доля обраних псевдоміток від загальної кількості псевдоміток, тобто алгоритм оптимізації псевдоміток видає $p \times 100\%$ найбільш прийнятних псевдоміток для подальшого тренування.

Псевдокод алгоритму наведено нижче. На вхід алгоритму подається нейронна мережа P(w), цільові зображення It, частка обраних псевдоміток від загальної їх кількості p.

```
for t=1 to T do
      P_It = P(w,It)
      MP_It = max(P_It, axis=0)
      M = [M, matrix_to_vector(MP_It)]
end
M = sort(M,order=descending)
len = length(M) × p
k = -log(M[len])
return k
```

При запропонованому підході з кожною ітерацією кількість псевдоміток, що використовується для навчання буде зростати. Якщо пропорція використаних псевдоміток для першої ітерації складатиме 20%, то з кожною ітерацією вона буде зростати на декілька процентів доки не досягне максимальної можливої пропорції 50%.

Параметр k_c відповідає параметру k, але враховує збалансованість класів. Алгоритм знаходження k_c описано нижче.

```
for t=1 to T do
      P_It = P(w,I[t])
      LP_It = argmax(P, axis=0)
      MP_It = max(P, axis=0)
      For c = 1 to C do
      MP_c_It = MP_It(LP_It == c)
      M_c = [M_c, matrix_to_vector(MP_c_It)]
end
for c = 1 to C do
      M_c = sort(M_c, order=descending)
      len = length(M_c) × p
      k_c = -log(M_c[len])
return k_c
```

Зауважимо, що параметр k_c знаходиться для кожного класу. Параметру k_c задається значення, при якому $\exp(-k_c)$ дорівнює вірогідності пікселя, який

розташований на місці $round(p * N_c)$, де N_c позначає кількість пікселів, які класифікуються як клас c. Як бачимо, ідея полягає у тому, щоб розрахувати граничну вірогідність коректності класифікації для псевдоміток кожного класу окремо. В інших аспектах фільтрація псевдоміток проводиться так само, як і при відсутності балансування класів.

3.3 Використання просторових подібностей між доменами при навчанні моделі

В деяких галузях аналізуючи зображення вихідного та цільового доменів можна зробити висновок, що деяки об’єкти розміщуються частіше в певних частинах зображення. Так, наприклад, розглядаючи датасети призначені для семантичної сегментації для самокерованих автомобілей, бачимо, що на всіх зображеннях дорога розміщується внизу зображення, а небо – вгорі. Схожий підхід, але на основі 3D даних було запропоновано у [30].

За умови, що цільовий та вихідний домени містять схожі зображення, вказані закономірності можуть бути використані при доменній адаптації. Додання цих даних у якості параметра до тренування повинно пришвидшити навчання моделі і поліпшити точність сегментації.

Для того, щоб внести ці дані до процесу навчання необхідно створити карти для кожного класу, що відображають частоту належності і-го пікселя до цього конкретного класу вздовж усіх зображень. Для цього підраховується частота появи класу для кожного пікселя, потім проводиться згладжування отриманої карти.

Введемо $q_n(c)$, що повертає частоту класу c для пікселя n. Після того, як отримані значення частот, необхідно провести нормалізацію таким чином, щоб виконувалася умова $\sum_{i=1}^{N} q_n(c) = 1$.

На рисунку 3.1 зображено карту частот класів, отриманих з датасету синтетичних зображень GTA5. Блакитним кольором позначені місця, де пікселі рідко відносяться до обраного класу, жовтим – місця зображення, де пікселі часто належать до цього класу.

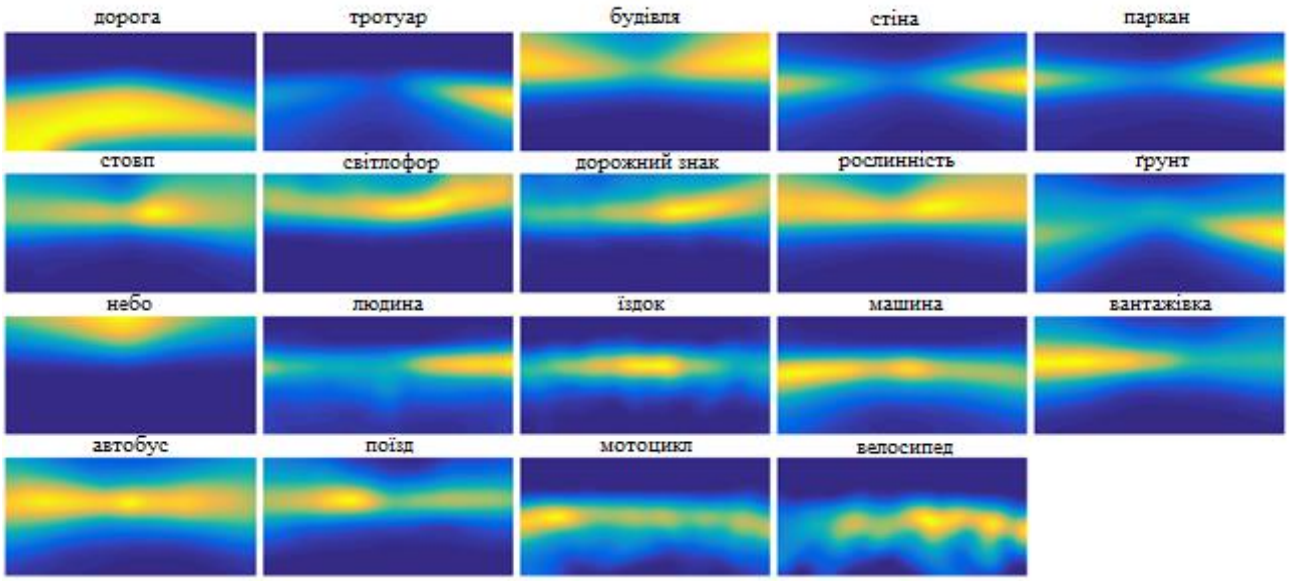

Рисунок 3.1 – Карта частот класів

Функція мінімізації з урахуванням частотних карт має вигляд:

$$\min_{w,\hat{y}} \zeta_{SP}(w,\hat{y}) = -\sum_{s=1}^{S}\sum_{n=1}^{N} y_{s,n}^{T} \log\left(p_n(w,I_s)\right)$$
$$-\sum_{t=1}^{T}\sum_{n=1}^{N}\sum_{c=1}^{C}[\hat{y}_{t,n}^{(c)} \log\left(q_n(c)p_n(c|w,I_t)\right) + k_c\hat{y}_{t,n}^{(c)}] \quad (3.9)$$
$$\hat{y}_{t,n} \in \{e|e \in \mathbb{R}^C\} \cup 0, \forall t,n;\ k_c > 0, \forall c$$

Щоб додати частотні карти до описаному вище методу, результат багатозмінної логістичної функції помножується на частотні карти.

3.4 Результати дослідження балансування класів

Для оцінки описаного підходу було проведено експеримент для адаптації домену на конволюційних мережах. В ході експерименту модель, натренована на датасеті SYNTHIA[31], була адаптована до датасету Cityscapes[32]. Модель – FCN8s-VGG16[33].

Результати, отримані методом, були порівняні з результатами, отриманими при застосуванні моделі на цільовому домені без адаптації, та результатами моделі, отриманими після тренування на датасеті Cityscapes. Валідаційний датасет Cityscapes використано у ролі цільового домену. Метрика – середнє відношення перетину до сукупності. Отримані результати наведені в таблиці 3.1.

Таблиця 3.1 – Результати досліджень для датасетів SYNTHIA та Cityscapes

Класи	Без адаптації	Запропонований метод	Еталон
Дорога	0.172	0.696	0.926
Тротуар	0.197	0.287	0.863
Будівля	0.473	0.695	0.784
Стіна	0.011	0.121	0.607
Паркан	0.0	0.001	0.852
Стовп	0.191	0.254	0.897
Світлофор	0.03	0.119	0.514
Дорожній знак	0.091	0.136	0.485
Рослинність	0.718	0.82	0.431
Небо	0.783	0.819	0.341
Людина	0.376	0.491	0.605
Їздок	0.047	0.145	0.738
Машина	0.422	0.66	0.596

Кінець таблиці 3.1

Автобус	0.09	0.066	0.194
Мотоцикл	0.001	0.037	0.417
Велосипед	0.009	0.324	0.558
mIoU	0.242	0.281	0.593

Оцінка адаптованої моделі має бути краще оцінки моделі натренованої виключно на вихідному датасеті (без адаптації). В ідеальному випадку адаптована модель продемонструє ефективність сегментації на рівні моделі, натренованої на цільовому датасеті (еталон).

За колонкою «Без адаптації» найгірше модель сегментує класи «стіна», «паркан», «світлофор», «мотоцикл» та «велосипед». Це обґрунтовується наступним: світлофори рідко зустрічаються в датасеті і у них відсутнє кольорове освітлення, паркани у SYNTHIA зовнішньо значно відрізняються від Cityscapes, мотоцикли та велосипеди відносяться до одного класу, а стіни до класу «будівлі».

За всіма класами крім класу «автобус» адаптована модель демонструє кращі показники. Результати адаптації для класу «велосипед» значно кращі ніж для класу «мотоцикл», бо в SYNTHIA на зображеннях велосипеди зустрічаються помітно частіше ніж мотоцикли.

В середньому адаптація моделі підвищила ефективність сегментації на 10%. Для класів, які були погано перенесені в новий домен, досягнуто суттєвих покращень.

Експеримент було повторно проведено для датасетів GTA[34] та Cityscapes. Цього разу додатково розглядається адаптація без застосування карт частот класів. Результати дослідження наведені в таблиці 3.2. Як бачимо, адаптація домену з балансуванням класів покращила сегментацію всього зображення в цілому і серед класів за винятком окремих класів. Погіршення результату для цих класів може бути пов’язано з використанням карт ознак.

Таблиця 3.2 – Результати досліджень для датасетів GTA та Cityscapes

Класи	Без адаптації	Балансування класів	Балансування класів + карти частот класів	Еталон
Дорога	0.44	0.567	0.581	0.896
Тротуар	0.121	0.168	0.308	0.589
Будівля	0.386	0.437	0.441	0.785
Стіна	0.133	0.148	0.183	0.514
Паркан	0.087	0.095	0.095	0.458
Стовп	0.199	0.283	0.272	0.414
Світлофор	0.155	0.259	0.286	0.482
Дорожній знак	0.059	0.101	0.141	0.392
Рослинність	0.449	0.455	0.464	0.837
Ґрунт	0.134	0.157	0.251	0.383
Небо	0.37	0.416	0.481	0.654
Людина	0.37	0.372	0.326	0.593
Їздок	0.103	0.062	0.145	0.376
Машина	0.312	0.319	0.414	0.833
Вантажівка	0.061	0.037	0.059	0.411
Автобус	0.012	0.022	0.125	0.485
Поїзд	0.018	0.054	0.012	0.204
Мотоцикл	0.108	0.189	0.14	0.353
Велосипед	0.029	0.324	0.286	0.417
mIoU	0.209	0.253	0.281	0.547

У порівнянні з моделлю, що не застосовує карти частот класів, результати погіршилися для класів «стовп», «поїзд», «мотоцикл», «велосипед», «людина». Це обґрунтовується тим, що розташування пікселів цих класів сильно відрізняється в GTA та Cityscapes. Порівняльний графік моделей наведено на рисунку 3.2.

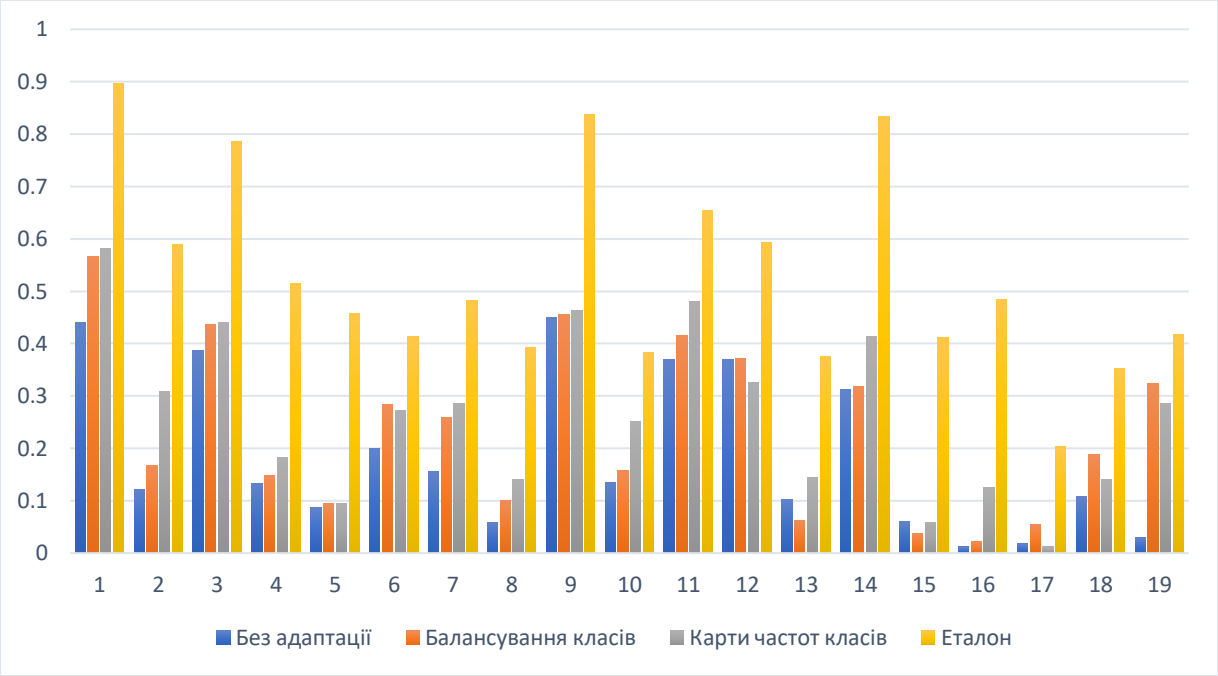

Рисунок 3.2 – Результати дослідження для GTA та Cityscapes

В подальших дослідженнях має сенс перевірити описаний підхід з іншими більш потужними моделями глибинного навчання, дослідити, які значення гіперпараметрів слід вводити у мережу і яким чином визначити, чи можна використовувати карти частот пікселів для двох довільних доменів.

Експерименти продемонстрували, що використання карт частот класів не гарантує покращення сегментації. Одним з сценаріїв, коли карти можуть призвести до погіршення результатів, – різні правила руху в доменах. Уявимо, що вихідний домен складається з міст з правостороннім рухом, а цільовий – міст з лівостороннім рухом. На картах, отриманих з вихідного датасету, буде помітна схильність деяких класів з’являтися у лівій частині зображення. В цьому можна переконатися, якщо подивитися на карти класів «машина», «вантажівка», «автобус» на рисунку 3.1. Деякі інші класи, наприклад «паркан», «стіна», «ґрунт» навпаки схильні частіше з’являтися в правій частині зображення. В цільовому домені з огляду на лівосторонній рух такі

закономірності появи класів не будуть виконуватися. З цього слід зробити висновок, що використання карт ознак потребує розуміння відмінностей між доменами.

Порівняння результатів роботи моделі з іншими рішеннями наведено в таблиці 3.3. Серед них повністю конволюційна мережа(FCN)[35] та модель доменної адаптації за навчальним планом(Curr. DA)[36] – архітектури, що складаються з однієї мережі, і дві генеративно-змагальні моделі: GAN DA[37] та MAA[38]. Усі методи окрім MAA використовують модель FCN8s-VGG16. Вихідний датасет – SYNTHIA, цільовий датасет – Cityscapes.

Таблиця 3.3 – Порівняння моделей доменної адаптації

Метод	Модель	mIoU
FCN	Dilitation-Frontend[39]	0.202
Curr. DA	FCN8s-VGG16	0.29
GAN DA	FCN8s-VGG16	0.348
MAA	DeepLab-v2	0.467
Запропонований метод	FCN8s-VGG16	0.281

З таблиці видно, що метод демонструє результати на тому ж рівні, що й інші одномережеві моделі, і поступає у точності генеративно-змагальним моделям. Втім, важливою перевагою запропонованого методу є швидше навчання у порівнянні з GAN DA та MAA.

Також бачимо, що MAA демонструє кращі результати ніж усі інші методи і при цьому дослідження проводилося на моделі DeepLab-v2[40], що загалом демонструє вищу точність ніж FCN8s-VGG16. Вірогідно, що при застосуванні розглянутого методу з DeepLab-v2, буде отримано кращі результати. Аналіз ефективності методу з різними архітектурами може бути предметом подальших досліджень.

ВИСНОВКИ

Незважаючи на великий темп зростання можливостей глибинного навчання, появи великої кількості датасетів та вдосконалення апаратних засобів для глибинного навчання, існує ряд задач, для яких навчання моделі з нуля неможливе або потребує багато зусиль. Одна з таких задач – семантична сегментація дорожніх зображень для різних місцевостей.

Оскільки для цієї задачі характерна відсутність розмічених даних, а створення анотацій займає багато часу, компромісне рішення полягає у адаптації існуючої моделі до нового домену.

В даній роботі запропоновано метод, який базується на генерації псевдоміток з цільового домену для розширення тренувального датасету. Хоча методи засновані на генерації псевдоміток часто використовуються при вирішенні задач адаптації домену, проблема балансування класів все ще залишається місцем для вдосконалення. Розроблений метод пропонує ввести ряд гіперпараметрів для нейронної мережі, які контролюватимуть рівномірну участь класів у налаштуванні моделі. Крім цього метод намагається використати дані про схожість двох доменів для адаптації моделі. Для цього вздовж всього вихідного датасету підраховується частота появи класів у кожному пікселі. Отримані карти частот пікселів використовуються у нейронній мережі фільтрування псевдоміток та покращеного балансування класів.

Метод був досліджений на датасетах GTA та Cityscapes. Результати роботи адаптованої моделі були проаналізовані у порівнянні з двома моделями: без адаптації та натренованою на цільовому домені. У порівнянні з моделлю без адаптації метод продемонстрував кращі результати сегментації вздовж усіх крім двох. При своїй відносній простоті метод довів свою ефективність, але погіршення результатів для двох класів вказують на необхідність подальшого дослідження карт частот класів і їхнього впливу на адаптацію домену.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Michael I Jordan and Tom M Mitchell. Machine learning: trends, perspectives,and prospects.Science, 349(6245):255–260, 2015.

2. Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Delving deep intorectifiers: surpassing human-level performance on imagenet classification. InInternational Conference on Computer Vision, pages 1026–1034, 2015.

3. Jianqiang Wang, Lei Zhang, Dezhao Zhang, and Keqiang Li. An adaptive longitudinal driving assistance system based on driver characteristics.IEEE Transactionson Intelligent Transportation Systems, 14(1):1–12, 2013.

4. Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. Bert: Pre-training of deep bidirectional transformers for language under-standing.arXiv preprint arXiv:1810.04805, 2018.

5. S. J. Pan and Q. Yang, “A survey on transfer learning,”Transactions on Knowledge and Data Engineering, vol. 22, no. 10,pp. 1345–1359, 2010.

6. K. Weiss, T. M. Khoshgoftaar, and D. Wang, “A survey of transfer learning,”Journal of Big Data, vol. 9, no. 3, 2016.

7. Bimbraw, Keshav. (2015). Autonomous Cars: Past, Present and Future - A Review of the Developments in the Last Century, the Present Scenario and the Expected Future of Autonomous Vehicle Technology. ICINCO 2015 - 12th International Conference on Informatics in Control, Automation and Robotics, Proceedings. 1. 191-198. 10.5220/0005540501910198.

8. Liu, Xiaolong, Zhidong Deng, and Yuhan Yang. “Recent Progress in Semantic Image Segmentation.” Artificial Intelligence Review 52.2 (2018): 1089–1106. Crossref. Web.

9. A. Yerokhin, O. Turuta, A. Babii and A. Nechyporenko, "Intelligent information system of heterogeneous medical data analysis," 2017 12th International Scientific and

Technical Conference on Computer Sciences and Information Technologies (CSIT), Lviv, 2017, pp. 332-335, doi: 10.1109/STC-CSIT.2017.8098798.

10. O. G. Avrunin, M. Y. Tymkovych, I. Y. Abdelhamid, N. O. Shushliapina, Y. V. Nosova and V. V. Semenets, "Features of Image Segmentation of the Upper Respiratory Tract for Planning of Rhinosurgical Surgery," 2019 IEEE 39th International Conference on Electronics and Nanotechnology (ELNANO), Kyiv, Ukraine, 2019, pp. 485-488, doi: 10.1109/ELNANO.2019.8783739.

11. Wurm, Michael & Stark, Thomas & Zhu, Xiao & Weigand, Matthias & Taubenböck, Hannes. “Semantic Segmentation of slums in satellite images using transfer learning on fully convolutional neural networks." ISPRS Journal of Photogrammetry and Remote Sensing. 150. 59-69. 10.1016/j.isprsjprs.2019.02.006.

12. Chen, Y.H., Chen, W.Y., Chen, Y.T., Tsai, B.C., Frank Wang, Y.C., Sun, M.:No more discrimination: Cross city adaptation of road scene segmenters. In: ICCV(2017)

13. M. Cordts, M. Omran, S. Ramos, T. Rehfeld, M. Enzweiler,R. Benenson, U. Franke, S. Roth, and B. Schiele.Thecityscapes dataset for semantic urban scene understanding. InCVPR. IEEE, 2016.

14. J. Xie, M. Kiefel, M.-T. Sun, and A. Geiger. Semantic in-stance annotation of street scenes by 3d to 2d label transfer.InCVPR. IEEE, 2016.

15. W. Shimoda and K. Yanai. Distinct class-specific saliencymaps for weakly supervised semantic segmentation.InECCV. Springer, 2016.

16. Krizhevsky, A. & Hinton, G.. (2009). Learning multiple layers of features from tiny images. Computer Science Department, University of Toronto, Tech. Rep. 1.

17. Simonyan, Karen & Zisserman, Andrew. (2014). Very Deep Convolutional Networks for Large-Scale Image Recognition. arXiv 1409.1556.

18. M. Long, J. Wang, G. Ding, J. Sun, and P. S. Yu, “Transfer joint matching for unsupervised domain adaptation,” inIEEEConference on Computer Vision and Pattern Recognition (CVPR), 2014.

19. W. Dai, Q. Yang, G.-R. Xue, and Y. Yu, “Boosting for transfer learning,” inInternational Conference on Machine Learning(ICML), 2010.

20. H. Daum´e III, “Frustratingly easy domain adaptation,”CoRR, vol. arXiv:0907.1815, 2009.

21. B. Fernando, A. Habrard, M. Sebban, and T. Tuytelaars, “Unsupervised visual domain adaptation using subspace align-ment,” inIEEE International Conference on Computer Vision (ICCV), 2013.

22. B. Chu, V. Madhavan, O. Beijbom, J. Hoffman, and T. Darrell, “Best practices for fine-tuning visual classifiers to new domains,” inECCV Workshop on Transferring and Adapting Source Knowledge in Computer Vision (TASK-CV), 2016.

23. Y. Y. Ganin, E. Ustinova, H. Ajakan, P. Germain, H. Larochelle, F. Laviolette, M. Marchand, and V. Lempitsky, “Domain-adversarial training of neural networks,”Journal of Machine Learning Research, 2016.

24. M.-Y. Liu and O. Tuzel, “Coupled generative adversarial networks,” inAnnual Conference on Neural Information Pro-cessing Systems (NIPS), 2016.

25. Mottaghi, Roozbeh & Chen, Xianjie & Liu, Xiaobai & Cho, Nam-Gyu & Lee, Seong-Whan & Fidler, Sanja & Urtasun, Raquel & Yuille, Alan. (2013). The Role of Context for Object Detection and Semantic Segmentation in the Wild. 10.13140/2.1.2577.6000.

26. Lin, Tsung-Yi & Maire, Michael & Belongie, Serge & Hays, James & Perona, Pietro & Ramanan, Deva & Dollár, Piotr & Zitnick, C.. (2014). Microsoft COCO: Common Objects in Context. 8693. 10.1007/978-3-319-10602-1_48.

27. M. Biasetton, U. Michieli, G. Agresti and P. Zanuttigh, "Unsupervised Domain Adaptation for Semantic Segmentation of Urban Scenes," 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW), Long Beach, CA, USA, 2019.

28. Ronneberger, O., Fischer, P., Brox, T.: U-net: Convolutional networks for biomed-ical image segmentation. In: MICCAI. LNCS, vol. 9351, pp. 234–241. Springer(2015).

29. K. He, G. Gkioxari, P. Dollar and R. Girshick, "Mask r-cnn", *ICCV*, pp. 2980-2988, 2017.

30. Silberman, N., Fergus, R.: Indoor scene segmentation using a structured light sensor. In: ICCV Workshop on 3D Representation and Recognition (2011).

31. Ros, G., Sellart, L., Materzynska, J., Vazquez, D., Lopez, A.M.: The synthiadataset: A large collection of synthetic images for semantic segmentation of ur-ban scenes. In: CVPR (2016).

32. Cordts, M., Omran, M., Ramos, S., Rehfeld, T., Enzweiler, M., Benenson, R.,Franke, U., Roth, S., Schiele, B.: The cityscapes dataset for semantic urban sceneunderstanding. In: CVPR (2016).

33. Long, J., Shelhamer, E., Darrell, T.: Fully convolutional networks for semanticsegmentation. In: CVPR (2015).

34. Richter, S.R., Vineet, V., Roth, S., Koltun, V.: Playing for data: Ground truth from computer games. In: ECCV (2016).

35. Hoffman, J., Wang, D., Yu, F., Darrell, T.: Fcns in the wild: Pixel-level adversarialand constraint-based adaptation. arXiv preprint arXiv:1612.02649 (2016).

36. Zhang, Y., David, P., Gong, B.: Curriculum domain adaptation for semantic segmentation of urban scenes. In: ICCV (2017).

37. Sankaranarayanan, S., Balaji, Y., Jain, A., Lim, S.N., Chellappa, R.: Unsupervised domain adaptation for semantic segmentation with gans. arXiv preprintarXiv:1711.06969 (2017).

38. Tsai, Y.H., Hung, W.C., Schulter, S., Sohn, K., Yang, M.H., Chandraker, M.:Learning to adapt structured output space for semantic segmentation. CVPR(2018).

39. Yu, F., Koltun, V.: Multi-scale context aggregation by dilated convolutions. In:ICLR (2016).

40. Chen, L.C., Papandreou, G., Kokkinos, I., Murphy, K., Yuille, A.L.: Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, andfully connected crfs. IEEE Trans. PAMI40(4), 834–848 (2018).