УДК 004.855 https://doi.org/10.15407/pp2020.02-03.352

ПОБУДОВА СЕМАНТИЧНОЇ МОДЕЛІ ЗОБРАЖЕННЯ З ВИКОРИСТАННЯМ МАШИННОГО НАВЧАННЯ НА БАЗІ ЗГОРТКОВИХ НЕЙРОННИХ МЕРЕЖ

П.І. Андон, А.М. Глибовець, В.В. Куриляк

У роботі описано основні напрямки досліджень у сфері побудови моделей автоматизації комп'ютерного розпізнавання сутності цифрового зображення. Введено поняття семантичної моделі зображення та описано реалізацію моделі машинного навчання для вирішення задачі автоматичної побудови такої моделі для вхідного зображення. Семантична модель складається зі списку об'єктів, які показано на зображенні, та їх зв'язків. Розроблена модель була порівняна з іншими рішеннями для цієї самої проблеми і показала кращі результати в усіх, за винятком одного, випадків. Ефективність роботи моделі обґрунтована використанням останніх досягнень машинного навчання, зокрема ZNM, TL, моделей Faster R-CNN і VGG16. Значна частина зв'язків представлених на зображенні є просторовими зв'язками, таким чином, для кращої роботи моделі, потрібно використовувати цей факт у її проектуванні, що і було зроблено.

Ключові слова: семантична модель зображення, машинне навчання, комп'ютерний зір, згорткові нейронні мережі, зв'язки на зображенні.

В работе описаны основные направления исследований в области построения моделей автоматизации компьютерного распознавания сущности цифрового изображения. Введено понятие семантической модели изображения и описано реализацию модели машинного обучения для решения задачи автоматического построения такой модели для входного изображения. Семантическая модель состоит из списка объектов, которые показаны на изображении, и их связей. Разработанная модель была сравнена с другими решениями для этой самой проблемы и показала лучшие результаты во всех, за исключением одного, случаев. Эффективность работы модели обоснована использованием последних достижений машинного обучения, в частности CHC, TL, моделей Faster R-CNN і VGG16. Значительная часть связей представленных на изображении есть пространственными связями, таким образом, для лучшей работы модели, нужно использовать этот факт в ее проектировании, что и было сделано.

Ключевые слова: семантическая модель изображения, машинное обучение, компьютерное зрение, сверточные нейронные сети, связи на изображении.

This paper describes the main areas of research in the field of developing computer models for the automatization of digital image recognition. The concept of the semantic image model is introduced and the implementation of the machine learning model for solving the problem of automatic construction of such a model is described. The semantic model consists of a list of objects represented in the image and their relationships. The developed model was compared to other solutions and showed better results in all but one case. The performance of the model is justified by the use of the latest achievements of machine learning, including ZNM, TL, Faster R-CNN, and VGG16. Much of the links represented in the image are spatial links, so for the model to work better, you need to use that fact in designing it, which was done.

Key words: semantic image model, machine learning, computer vision, convolutional neural networks, image links.

Вступ

Розуміння (інтерпретація) зображень – одна з найактуальніших задач штучного інтелекту [1]. Зацікавлення багатьох дослідників обумовлено великим спектром можливих застосувань. Візуальна інформація є одним із найбільш затребуваним і поширеним, зокрема в Інтернеті, типом інформації. Зі збільшенням її кількості, питання про автоматизацію обробки стає надзвичайно актуальним. Прості задачі, такі як автоматичне анотування зображень або пошук схожих зображень, можна вважати вирішеними [2–5], але в плані справді глибокого розуміння зображеного, сучасні методи далекі від ідеалу.

За останні роки зроблено багато досліджень для зближення рівня розуміння зображень людьми і автоматизованими системами, зокрема, завдячуючи вирішенню таких задач, як створення текстового опису або пошук об'єктів на зображенні [1, 5]. Проте, для повноцінного тлумачення («розуміння») змісту зображення (того, що представлено на зображенні) комп'ютером потрібно отримати формальне структуроване представлення усієї інформації, що розміщена на зображенні. Формат і структура такого представлення потребує дослідження. Ми називаємо таке задання ***семантичною моделлю зображення.*** Семантична модель складається зі списку об'єктів, які показані на зображенні, та їх зв'язків.

Отже, метою цієї роботи є висвітлення нашого бачення автоматизованої побудови опису зображення. Ми вводимо термін «*семантична модель*» для формалізації представлення зображеного на картинці. Присутність терміну «*семантична*» означає, що таке представлення має допомагати розкривати сутність зображеного на основі підходу схожого на механізм інтерпретації зображення людиною, а «модель» вимагає чіткості і формалізації структури представлення. Ці зауваги важливі, оскільки ми маємо за мету автоматизацію процесу комп'ютерної обробки зображень.

Існує кілька підходів до вирішення задачі «розуміння зображень» або пов'язані з нею частково.

Серед основних напрямків досліджень у сфері побудови моделей автоматизації комп'ютерного розпізнавання сутності цифрового зображення виділяють споріднені задачі класифікації зображень [6], текстового опису зображень [7], визначенні зв'язків на зображенні [8]. В останніх дослідженнях акцент зроблено на висвітлення перспективних підходів визначення зв'язків на зображенні з використанням візуальних фраз «Recognition Using Visual Phrases» [9], «Visual Relationship Detection With Language Priors» [10], «Deep Relational Network» [8]. Остання модель розроблена спеціально для ефективного використання статистичних характеристик залежності між об'єктами і їх зв'язками на зображенні у машинному навчанні на базі згорткових нейронних мереж.

У цій роботі ми висвітлимо власний підхід до вирішення поставленої проблеми побудови семантичної моделі зображення. Модель орієнтована на машинне навчання (МН). Вона має отримувати на вхід зображення, а на виході повертати множину суб'єктів, об'єктів й їх зв'язків.

Розробка семантичної моделі

У МН, експертні знання про предметну область застосування дозволяють краще зрозуміти проблему та отримати оптимальніше рішення. Тому проектування моделей починається з аналізу даних і пошуку потрібних колекцій. Ми теж почнемо з пошуку потрібної колекції даних.

Навчальна вибірка. Для вирішення нашої задачі ми скористалися колекцією Visual Genome [11]. Вона створена та успішно використовується для вирішення різного роду задач комп'ютерного зору, пов'язаних з покращенням «розуміння» зображення. На офіційному сайті [12] можна знайти навчальні колекції, зокрема для задач пошуку об'єктів, взаємозв'язків, створення текстового опису. У ній міститься близько 108 000 зображень. Для кожного з яких вказані об'єкти, їх позиції і зв'язки між кожним з них. Тому, ці дані – якраз те, що потрібно для вирішення нашої задачі.

Формат даних Visual Genome дещо надлишковий, містить багато шуму і, в цілому, з ним не так просто працювати. Тому багато дослідників використовують модифікації цієї вибірки або перетворюють її в інший формат. Зокрема, ми використовували формат запропонований в [8]. У ньому всі дані відразу розбиті на навчальну і тестову вибірку. Списки класів об'єктів і зв'язків подаються окремо. Для кожного зображення є його розташування (img_path); класи об'єктів, які зображені на ньому (classes); позиції об'єктів, які виступають в ролі суб'єкта (ix2) та об'єкта (ix2); обмежувальні прямокутники для кожного локалізованого об'єкта (boxes); і для кожної пари суб'єкт-об'єкт клас зв'язку між ними (rel_classes). Цю інформацію можна візуалізувати, відобразивши зображення, всі його локалізовані об'єкти з їх класами і окремо вказати всі зв'язки між ними.

Згорткові нейронні мережі. В цій роботі ми будемо використовувати згорткові нейронні мережі (ЗНМ) з використанням **transfer learning**. Вони були розроблені для оптимального вирішення задач обробки зображень використовуючи методи нейронних мереж.

Моделлю, якою ми скористалися у цій роботі є VGG16 [3]. Це ЗНМ, яка розроблена для вирішення задачі класифікації колекції ImageNet в 1000 класів. Вона показала тільки 7.5 % помилки. В основному досягнення цього результату забезпечило саме використання глибокої архітектури. Вона містить 16 прихованих шарів, що, навіть по сьогоднішніх мірках, багато. З часом, VGG16 показала себе здатною до простого трансформування під нові задачі і стала де-факто стандартом для TL у вирішенні задач комп'ютерного зору. Після VGG16 запропоновано кілька інших мереж, які показували навіть кращі результати для цієї самої задачі, але вони користуються меншим успіхом для TL, частково через те, що інтерпретація їх результатів іноді мають значну складність.

VGG16 включена в багато бібліотек і фреймворків машинного навчання, що значно полегшує її застосування. Для завантаження даних при роботі з моделями машинного навчання в PyTorch існує спеціальний механізм, який полягає в оголошенні класу Dataset [13]. Саме він використовувався нами для завантаження даних у нашій розробці. Крім діставання самих даних ще відбувається нормалізація зображень. Це необхідний елемент для роботи з моделлю [14].

Для перевірки роботи моделі, можна передати їй якесь зображення, наприклад з навчальної вибірки Visual Genome. Для цього потрібно перевести модель у режим оцінки. Результатом роботи моделі буде ймовірність належності зображення на картинці визначеному класу зображень. Відповідно, потрібно отримати позицію найбільшого елемента і знайти, який саме клас відповідає знайденому індексу.

Оскільки модель використовувалась для класифікації зображень, ми отримали одне значення, а не кілька, як у навчальній вибірці з Visual Genome. Також варто зазначити, що VGG16 використовує інший набір класів (класи ImageNet), тому для того, щоб отримувати класи об'єктів, які були у навчальній вибірці його доведеться перетренувати.

Додавання RoI Pool шару. Першою нашою модифікацією моделі VGG16 було додавання RoI Pool шару. RoI Pool (шар діставання регіонів зацікавленості) — це Pool-шар ЗНМ фіксованого розміру (є параметром шару) для певного регіону попереднього шару ЗНМ [5]. Він запропонований в моделі Faster R-CNN [15].

Цей шар дозволяє вибирати окремі риси з певного регіону шару і так явно локалізувати об'єкти. В Faster R-CNN він використовувався для виявлення об'єктів. Нам потрібно передбачити класи зв'язків, тому

потенційно корисними можуть бути регіони, в яких виявлені ці зв'язки. Виникає питання, яким чином визначити, що зв'язок лежить у певному регіоні? Для того, щоб отримати відповідь, спочатку можна розглянути простішу задачу.

Маючи позиції об'єктів (з вхідних даних навчальної вибірки), можемо застосувати RoI Pool шар до них. Оскільки для ідентифікації зображення можемо використати фіксовану кількість об'єктів, скажімо k, то застосування цього шару призведе до отримання k наборів зображення. Отже, якщо раніше модель повертала клас одного об'єкта, тепер ми можемо повертати класи k об'єктів.

Модель VGG16 повертає результат у вигляді матриці розміру (1, 1000), оскільки є 1000 класів для яких вона була натренована. Для кожного з цих класів, ми отримуємо ймовірність того, що цей клас є класом об'єкта, який міститься на зображенні. Тепер, повертаючи k класів, результат буде мати розмірність (k, 1000). Проте, нам потрібно передбачати класи з навчальної вибірки, а не класи ImageNet. Тому в моделі потрібно замінити останній повно-зв'язний шар на шар, який повертатиме правильну кількість класів і перетренувати модель. Варто зазначити, що всі ваги моделі, крім ваг цього останнього шару, ми залишаємо такими, якими вони були в натренованій моделі VGG16 і не будемо оптимізувати їх (проводити зворотнє поширення помилки), оскільки ці ваги відповідають шарам, які не пов'язані безпосередньо з класифікацією, а відповідають за визначення рис (feature extraction).

Ми замінили існуючий Adaptive Average Pool шар на RoI Pooling шар і таким чином, більшість елементів моделі залишаться незмінними. Для того, щоб не потрібно було змінювати FC-шари, які йдуть після Pool-шару, ми також використовували такий самий розмір шаблону як в Pool-шарі (7x7). Все це дозволило з невеликими зусиллями отримати нову модель для вирішення нашої задачі.

Оскільки ми робимо зміни у проході вперед (forward pass) у мережі, довелося створити нову модель і перевизначити в ній метод forward. Код цієї моделі наведено в лістингу 1.

Лістинг 1. Мережа з використанням VGG16 за допомогою TL

```
1  class Net(nn.Module):
2      def __init__(self, obj_num):
3          super(Net, self).__init__()
4
5          tl_model = vgg16(pretrained=True)
6          for param in tl_model.parameters():
7              param.requires_grad = False
8
9          self.features = tl_model.features
10         self.roi_pool = RoIPool((7, 7), 1 / 16.0)
11
12         classifier = list(tl_model.classifier.children())[:-1]
13         classifier.extend([nn.Linear(4096, obj_num)])
14         self.classifier = nn.Sequential(*classifier)
15
16     def forward(self, x, boxes):
17         x = self.features(x)
18         x = self.roi_pool(x, boxes)
19         x = x.view(x.size(0), -1)
20         x = self.classifier(x)
21         return x
```

Для роботи моделі спочатку потрібно створити всі шари, щоб потім використовувати їх при проході вперед. Тому в конструкторі ми спочатку отримуємо модель VGG16 з натренованими вагами (рядок 5) і вимикаємо поширення градієнта для всіх її параметрів (рядки 6, 7). Набір шарів features ми залишаємо незмінним (рядок 9), додаємо новий RoI Pool шар з необхідним розміром шаблону (рядок 10). У наборі шарів classifier змінюємо тільки останній шар (рядки 12–14). Змінений шар буде мати кількість вихідних шарів рівну кількості класів об'єктів навчальної вибірки (obj_num), яка визначатиметься в конструкторі. Варто зазначити, що останній шар набору classifier – єдиний, в якого значення для параметрів requires_grad буде рівним true (це значення за замовчуванням), тому це буде єдиний шар, для якого проводитиметься оптимізація ваг.

Крім того, потрібно також модифікувати клас для завантаження навчальної вибірки, оскільки тепер потрібно ще повертати позиції об'єктів (обмежувальні прямокутники). Його змінена форма наведена в лістингу 2.

Лістинг 2. Оновлений клас для завантаження даних, після змін в VGG16

```
1  class IsmDataset(Dataset):
2
3      def __init__(self, data):
4          self.data = data
5          self.img_transform = Compose([
6              ToTensor(),
7              Normalize([0.485, 0.456, 0.406],
8                        [0.229, 0.224, 0.225])
9          ])
10
11     def __len__(self):
12         return len(self.data)
13
14     def __getitem__(self, index):
15         data = self.data[index]
16
17         if (data == None):
18             return ()
19
20         file_path = data['img_path']
21
22         if (not os.path.exists(file_path)):
23             return ()
24
25         with open(file_path, 'rb') as f:
26             img = Image.open(f)
27             img_tensor = self.img_transform(img)
28
29         objects = data['classes'].astype(np.int64)
30
31         raw_boxes = data['boxes']
32         boxes = np.zeros((raw_boxes.shape[0], 5), dtype=np.float32)
33         boxes[:, 1:5] = raw_boxes
34
35         return img_tensor, boxes, objects
```

В рядках 31–33 потрібно було додати 0 до кожного списку обмежувальних прямокутників. Це пов’язано з тим, як RoIPool шар працює з позиціями об’єктів.

Вся навчальна вибірка розбивалася на 2 набори: навчальний і тренувальний. Для цього використовувався визначений у лістингу 8 клас, а також клас DataLoader. Код класу наведено в лістингу 3.

Лістинг 3. Поділ навчальної вибірки на тренувальну у навчальну

```
1  data_sets = { x: IsmDataset(read_dataset(x)) for x in ['train', 'test'] }
2  data_loaders = { x: DataLoader(data_sets[x]) for x in ['train', 'test'] }
```

Маючи всі необхідні елементи, можна запустити процес навчання. Для цього було визначено функцію, яка може одночасно займатись навчанням і тестуванням залежно від того, яку частину навчальної вибірки вона отримує. Цей підхід було адаптовано з офіційних рекомендацій [16].

У функції обчислювалися отримані значення з проходу мережі вперед і втрати, робилася зворотна пропозиція втрат і оптимізувалися параметри. Решта коду відповідала за перевірки фази тренування, збереження часу виконання, накопичення втрат та збереження моделі, що показала найкращий результат.

Функція має кілька параметрів, про які ще не йшла мова. Перший з них це criterion — критерій, тобто функція втрат. Оскільки нам потрібно передбачати кілька класів, ми використостовували MultiLabelMarginLoss — це функція втрат на основі SVM, але для кількох класів. Optimizer і scheduler — це об’єкти, які займатимуться покращенням значень ваг мережі після зворотнього поширення помилки. Optimizer додавав до ваг значення градієнта з певним кроком, а scheduler змінював крок навчання (learning rate) для підвищення ефективності навчання. Код для запуску навчання наведено в лістингу 4.

Лістинг 4. Запуск навчання моделі

```
net = Net(len(obj_classes)).to(device)
criterion = nn.MultiLabelMarginLoss()
last_layer_params = list(net.classifier.children())[-1].parameters()
optimizer = optim.Adam(last_layer_params, lr=0.001, momentum=0.9)
scheduler = lr_scheduler.StepLR(optimizer, step_size=7, gamma=0.1)
```

Додавання решти шарів. Повертаючись до питання отримання класів для зв'язків, потрібно узагальнити те, яка інформація у нас є на вході, що ми на даний момент маємо під час роботи мережі і який має бути вихідний результат.

Оскільки, на вхід моделі ми отримуємо об'єкти, а на виході хочемо отримати класи зв'язків між ними, вихідним форматом даних може бути зв'язок між кожною парою об'єктів. Для того, щоб ми могли проводити оптимізацію, нам потрібно закодувати класи зв'язків в one-hot вектор. Тому, якщо у нас є k об'єктів і n можливих класів зв'язків, результатом буде матриця розмірності k*(k–1) x n. Перевага цього способу полягає у отриманні знову задачі багатокласової класифікації, а, отже, можемо використати напрацювання з попереднього підрозділу.

Разом з тим, є деякі складнощі. Якщо для класифікації об'єктів, ми використовували RoI Pool шар для отримання візуальних рис для кожного об'єкта, тепер нам потрібні візуальні риси для кожного можливого зв'язку між об'єктами. Для вирішення цієї проблеми будемо використовувати об'єднані риси суб'єкта і об'єкта. Для їх отримання потрібно знайти об'єднання обмежувальних прямокутників суб'єкта та об'єкта. Це проста операція, її потрібно виконати для кожної пари об'єктів (крім самого себе, об'єкт в навчальній вибірці не може мати зв'язку сам із собою) і так отримаємо обмежувальні прямокутники для кожного зв'язку, який потрібно передбачити.

Слідуючи архітектурі VGG16, після Pool-шару мають йти 3 FC-шари, для того, щоб натренувати, власне, класифікацію зв'язків. Після цього можна додати останній FC-шар, у якого вихідна розмірність буде рівна кількості класів зв'язків і ми отримаємо знову багатокласовий класифікатор.

Роботу запропонованої моделі можна покращити, якщо крім візуальних рис, використовувати ще просторові. Цю ідею ми запозичили з раніше розглянутої роботи [8] і адаптували до нашого рішення. Додавання просторової інформації до роботи мережі допомагає, оскільки значна частина зв'язків є саме просторовими (під, на, біля, і т. п.) і явно виділяючи їх, ми спрощуємо їх правильне визначення. Та визначити ці види зв'язків маючи тільки візуальні риси об'єднаного регіону суб'єкта і об'єкта досить складно. Тому, використовуючи новий вид інформації, ми покращуємо роботу моделі.

Існує кілька способів представити просторові зв'язки між об'єктами, але для нас найбільш підходящим було представлення у вигляді масок. Для об'єкта і суб'єкта створюється маска зображення, де тільки пікселі, які входять в обмежувальний прямокутник кожного з них мають значення 1, решта дорівнюють 0. Отримані дві маски і є представленням просторових зв'язків. Перевага використання такого методу полягає у простій можливості представлення цих масок у вигляді вхідного шару для ЗНМ і скористатися сильною стороною ЗНМ для тренування визначення просторової залежності між ними. Оскільки вхідний шар буде складатись з обох масок, глибина шару буде 2 і всі згорткові операції будуть проходити на обох шарах. Зазначимо, що буде враховуватися і той факт, що обидва мають спільну область.

Для проектування згорткових шарів, які будуть використовуватись для просторових рис, ми теж за основу взяли згорткові шари VGG16, їх було 3, а після них йшов один FC-шар.

Тепер, маючи два види рис, потрібно якимось чином їх об'єднати. Стандартне рішення в такому випадку – додати ще один повно-зв'язний шар, який буде приймати обидва набори рис і об'єднувати їх. Це означає, що перед останнім FC-шаром, який згадувався раніше буде ще один FC-шар, який буде займатись об'єднанням результатів шарів для візуальних і просторових рисунків.

Для навчання фінальної моделі було використано той же код, який використовувався для навчання проміжної моделі.

Результати

Для оцінки якості роботи розробленої моделі, ми використовували метрику «повнота з K». Як уже згадувалось, вона була запропонована в [10] і стала стандартною для цього класу задач. Вона визначає частку правильно передбачених значень серед найкращих K передбачень. В основному, для K використовується значення 50 і 100. Таким чином ми отримуємо дві метрики: повнота з 50 і 100.

Розроблена модель приймає на вхід об'єкти, їх позиції і повертає зв'язки між ними. Для того, щоб оцінити її якість по згаданих метриках, нам потрібно використовувати тестовий набір даних з колекції Visual Genome. Проте, більш цікавою є задача отримання і об'єктів і зв'язків між ними тільки по вхідному

зображенню. Крім того, це одна з вимог, які ми ставили перед розробкою моделі.

Для вирішення цієї проблеми, ми скористались існуючою моделлю для пошуку об'єктів і їх позицій, а вже результат цієї моделі слугував вхідними даними до нашої моделі. Отже, робота моделі складалась з двох етапів. Аналогічний підхід використовувався і в ряді інших робіт [8, 9].

Як модель для пошуку об'єктів, ми використовували Faster R-CNN, яка уже згадувалась раніше. На даний час, вона досягає state-of-the-art результатів для вирішення поставленої задачі [15].

Таким чином, замість однієї задачі, з'являються дві. Перша – передбачення зв'язків між наперед заданими об'єктами. Друга – передбачення наборів суб'єкт-об'єкт-зв'язок. Ці задачі називаються визначення предикатів (Prediction Detection) і передбачення зв'язків (Relationship Prediction) відповідно. Деякі роботи пропонували свої рішення цих проблем, ми порівняємо наші результати з ними. На рисунку показано графіки залежності обраних метрик від кількості ітерації для кожної з задач відповідно. Можна бачити, що в обох випадках, найвище значення було на 5-ій ітерації і після цього модель починала перенавчатись. В табл. 1 наведено порівняння найкращих результатів роботи нашої моделі на тестовому наборі з результатами досліджень інших авторів.

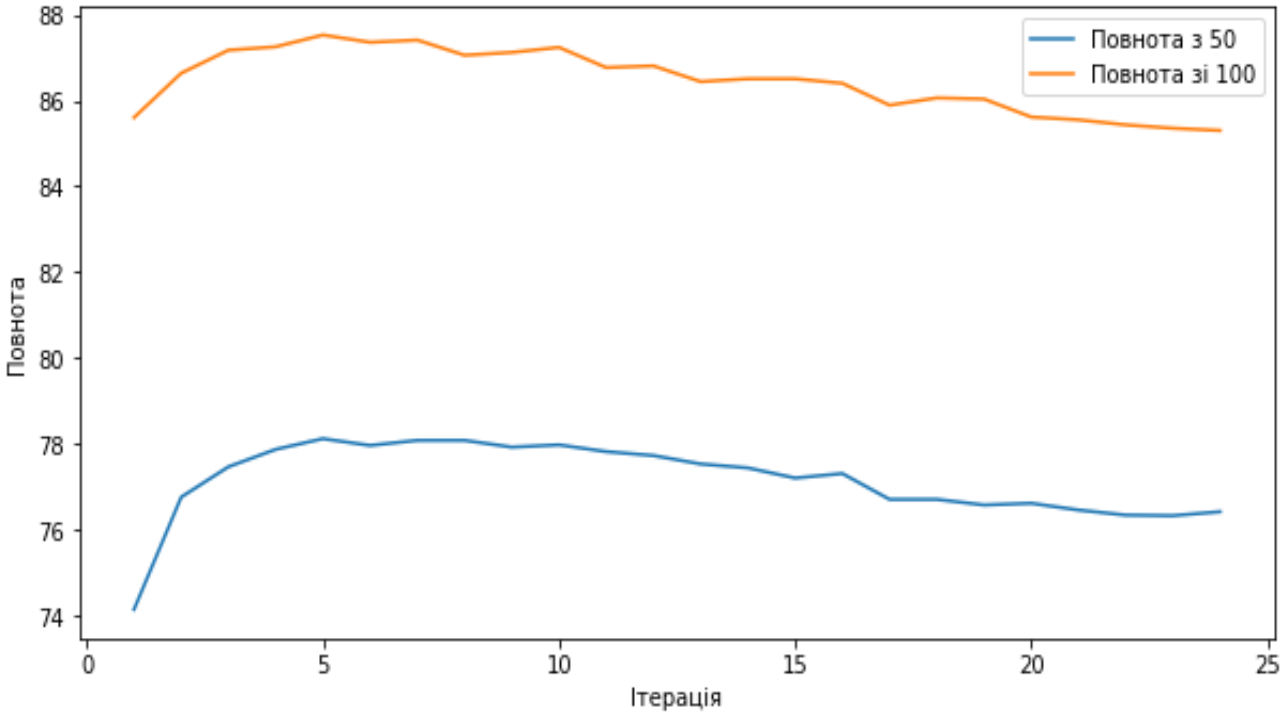

Рисунок. Залежність повноти від ітерацій задачі визначення предикатів

Таблиця 1. Порівняння результатів роботи створеної моделі

Модель	Визначення предикатів		Передбачення зв'язків	
	Повнота з 50	Повнота зі 100	Повнота з 50	Повнота зі 100
[16] 2015	0.97	1.91	–	–
[4] 2016	47.87	47.87	13.86	14.70
[13] 2018	80.78	81.90	17.73	20.88
Наша	78.12	87.53	18.56	22.05

Для того, щоб краще зрозуміти, які саме результати повертає розроблена модель та проаналізувати їх особливості, в табл. 2 наведено кілька результатів згенерованих розробленою моделлю. Для кожного зображення, ми пропонуємо як результат згенерований з відомими об'єктами (визначення предикатів), так і без них (передбачення зв'язків). Для другого випадку, спочатку потрібно було вхідне зображення передати детектору Faster R-CNN, а вже його результат передати в модель.

Для кожного зображення, ми обрали 20 передбачених триплетів з найвищими значенням функції оцінки.

Таблиця 2. Деякі приклади роботи моделі

Зображення	Визначення предикатів	Передбачення зв'язків
1	**2**	**3**
	sunglasses - on - person watch - on - person jeans - on - person building - behind - dog person - wear - jeans person - wear - sunglasses person - wear - watch building - behind - person glasses - on - person dog - in the front of - building person - wear - glasses person - in the front of - building bag1 - behind - glasses bag - behind - glasses glasses - next to - bag glasses - next to - bag1 person - hold - bag watch - near - glasses person - hold - bag1 bag1 - next to - dog	shirt - on - person shirt - on - person2 shirt - on - person1 jeans - on - person2 glasses - on - person2 jeans - on - person glasses - on - person jeans - on - person1 glasses - on - person1 jacket - on - person2 jacket - on - person1 person1 - wear - jacket person2 - wear - jacket jacket - on - person person - wear - jacket person1 - wear - jeans person1 - wear - shirt person - wear - jeans person2 - wear - shirt person2 - wear - jeans
	shirt - on - person jeans - on - person shorts - on - person person - wear - jeans person - wear - shorts person - wear - shirt shirt - above - shorts cabinet - above - bag shirt - above - jeans cabinet - behind - person shorts - below - shirt shirt - behind - bag bag - above - shorts person - hold - bag jeans - below - shirt bag - above - jeans shorts - next to - bag person - in the front of - cabinet bag - below - cabinet bag - near - shirt	shirt - on - person1 shirt1 - on - person shirt - on - person shirt1 - on - person1 pants - on - person1 pants - on - person person - wear - shirt person1 - wear - shirt person - wear - shirt1 person1 - wear - shirt1 person1 - wear - pants person - wear - pants shirt - above - pants shirt1 - above - pants pants - below - shirt pants - below - shirt1 person - next to - person1 person1 - next to - person shirt - behind - shirt1 shirt1 - behind - shirt

Закінчення табл. 2

1	2	3
	jeans - on - chair2 jeans - on - chair1 jeans - on - chair chair1 - next to - chair chair1 - next to - chair2 chair2 - next to - chair chair - next to - chair2 chair - next to - chair1 chair2 - next to - chair1 bed - near - chair bed - near - chair2 bed - near - chair1 chair1 - near - bed chair - near - bed chair2 - near - bed bed - has - jeans jeans - on - bed chair2 - has - jeans chair - has - jeans chair1 - has - jeans	plant - on - table1 plant - on - table lamp - on - table1 table - on - street table1 - on - street chair - on - street table1 - has - plant lamp - on - table chair - next to - table1 chair - next to - table table - has - plant lamp - behind - plant lamp - next to - chair street - below - lamp street - under - table street - under - table1 plant - next to - lamp table1 - has - lamp street - under - chair table - has - lamp
	shoes - on - person person - wear - jeans person - wear - hat person - wear - shoes jeans - on - person hat - above - shoes hat - on - person jeans - above - shoes roof - above - person hat - over - jeans jeans - near - hat shoes - beneath - jeans person - on - elephant shoes - on the right of - hat roof - above - shoes person - under - roof elephant - next to - person roof - above - jeans shoes - on - elephant roof - above - elephant	shirt1 - on - person1 shirt1 - on - person3 shirt - on - person2 shirt - on - person3 shirt - on - person1 shirt1 - on - person2 shirt - on - person trees - behind - elephant trees - behind - person trees - behind - person2 trees - behind - person3 person2 - wear - shirt1 person - wear - shirt building - behind - person2 shirt1 - on - person building - behind - person person3 - wear - shirt1 person2 - wear - shirt person - wear - shirt1 building - behind - person

Для першого зображення, можна бачити, що передбачення для першої задачі були отримані кращі. Вони більш детально описують зображення і містять різноманітні зв'язки, такі як «hold», «wear», «in front of», «next to», водночас, як у другої задачі є тільки «on» і «wear». Використання детектора, замість даних напряму з навчальної вибірки, очевидно, негативно впливає на результати.

Разом з тим, іноді такі прості передбачення можуть слугувати показником хорошої якості, як наприклад у зображеннях 2 і 3. Тут обидві задачі були вирішені якісно, хоча, в основному, вони теж використовують тільки просторові зв'язки і предикат «wear». Те, які види зв'язків будуть отримані залежить від виду зображення. Взагалі, це ще проблема самої навчальної колекції. Саме зв'язки «on» і «wear» були найпопулярнішими в навчальній вибірці, тому модель схильна передбачати їх. Іншим фактором є те, що ми явно використовували просторові риси в моделі, тому те, що значна частина передбачень саме пов'язана з позиціями об'єктів цілком очікувана.

Щодо третього зображення, хочеться ще додати що розроблена модель вміє розрізняти кілька об'єктів одного класу на зображенні. Зокрема, можна бачити, що вона успішно знайшла зв'язки для трьох крісел.

На останньому зображенні важливою є різниця між передбаченнями для різних задач. У випадку визначення предикатів, зосередження уваги йде на людину, яка сидить на слоні і все, що її оточує; водночас, як результати для передбачення зв'язків навіть не містять слона, зате описують кількох людей на задньому плані. Це тільки ще раз доводить, що для другої задачі, якість сильно залежить від якості роботи детектора.

Висновки

У роботі введено поняття семантичної моделі зображення та описано реалізацію моделі машинного навчання для вирішення задачі автоматичної побудови такої моделі для вхідного зображення. Семантична модель складається зі списку об'єктів, які представлені на зображенні, та їх зв'язків.

Крім того, в роботі розглядалися інші можливі та існуючі підходи для представлення семантичної моделі. Аналіз цих підходів показав, що саме підхід на основі зв'язків є найбільш детальним та придатним для вирішення різного роду задач, пов'язаних з розумінням зображення.

Розроблена модель була порівняна з іншими рішеннями для цієї самої проблеми і показала кращі результати в усіх, за винятком одного, випадків. Ефективність роботи моделі обґрунтована використанням останніх досягнень машинного навчання, зокрема ЗНМ, TL, моделей Faster R-CNN і VGG16.

З результатів роботи моделі, можна зробити висновок, що якість роботи для такого виду задач залежить від якості навчальної вибірки. Чим більш різноманітна і повна вона буде, тим кращі будуть результати. Крім того, дуже значна частина зв'язків представлених на зображенні є просторовими зв'язками, таким чином, для кращої роботи моделі, потрібно використовувати цей факт в її проектуванні, що і було зроблено.

Використання архітектури з кількома етапи призводить до того, що якість роботи останніх етапів сильно залежить від якості роботи перших. Так було у моделі при використанні детектора Faster R-CNN. З цього можна зробити висновок, що для покращення результатів роботи, треба займатись покращенням якості вирішення задачі детекції об'єктів.

Література

1. Karpathy A., Li Fei-Fei. Deep Visual-Semantic Alignments for Generating Image Descriptions [Електронний ресурс]. Режим доступу: https://cs.stanford.edu/people/karpathy/deepimagesent/
2. A visual proof that neural nets can compute any function [Електронний ресурс]. Режим доступу: http://neuralnetworksanddeeplearning.com/chap4.html
3. Simonyan K., Zisserman A. Very Deep Convolutional Networks for Large-Scale Image Recognition [Електронний ресурс]. Режим доступу: https://arxiv.org/pdf/1409.1556.pdf
4. Image Captioning [Електронний ресурс]. Режим доступу: http://shikib.com/captioning.html
5. Dai J. R-FCN: Object Detection via Region-based Fully Convolutional Networks [Електронний ресурс]. Режим доступу: https://arxiv.org/pdf/1605.06409.pdf
6. VGG16 – Convolutional Network for Classification and Detection [Електронний ресурс]. Режим доступу: https://neurohive.io/en/popular-networks/vgg16/
7. Vinyals O. Show and Tell: A Neural Image Caption Generator [Електронний ресурс]. Режим доступу: https://arxiv.org/pdf/1411.4555.pdf
8. Dai B. Detecting Visual Relationships with Deep Relational Networks [Електронний ресурс]. Режим доступу: https://arxiv.org/pdf/1704.03114.pdf
9. Sadeghi M. Recognition Using Visual Phrases [Електронний ресурс]. Режим доступу: http://vision.cs.uiuc.edu/phrasal/recognition_using_visual_phrases.pdf
10. Lu C. Visual Relationship Detection with Language Priors [Електронний ресурс]. Режим доступу: https://arxiv.org/pdf/1608.00187.pdf
11. Krishna R. Visual Genome Connecting Language and Vision Using Crowdsourced Dense Image Annotations [Електронний ресурс]. Режим доступу: https://arxiv.org/pdf/1602.07332.pdf
12. Visual Genome [Електронний ресурс]. - Режим доступу: https://visualgenome.org
13. Data Loading and Processing Tutorial [Електронний ресурс]. Режим доступу: https://pytorch.org/tutorials/beginner/data_loading_tutorial.html
14. TorchVision Models [Електронний ресурс]. Режим доступу: https://pytorch.org/docs/stable/torchvision/models.html
15. Ren S. Faster R-CNN: Towards Real-Time Object Detectionwith Region Proposal Networks [Електронний ресурс]. Режим доступу: https://papers.nips.cc/paper/5638-faster-r-cnn-towards-real-time-object-detection-with-region-proposal-networks.pdf

16. Chilamkurthy S. Transfer Learning Tutorial [Електронний ресурс]. Режим доступу: https://pytorch.org/tutorials/beginner/transfer_learning_tutorial.html

References

1. Karpathy A., Li Fei-Fei. Deep Visual-Semantic Alignments for Generating Image Descriptions [Electronic resourse]. Mode of access: https://cs.stanford.edu/people/karpathy/deepimagesent/
2. A visual proof that neural nets can compute any function [Electronic resourse]. Mode of access: http://neuralnetworksanddeeplearning.com/chap4.html
3. Simonyan K., Zisserman A.Very Deep Convolutional Networks for Large-Scale Image Recognition [Electronic resourse]. Mode of access: https://arxiv.org/pdf/1409.1556.pdf
4. Image Captioning [Electronic resourse]. Mode of access: http://shikib.com/captioning.html
5. Dai J. R-FCN: Object Detection via Region-based Fully Convolutional Networks [Electronic resourse]. Mode of access: https://arxiv.org/pdf/1605.06409.pdf
6. VGG16 – Convolutional Network for Classification and Detection [Electronic resourse]. Mode of access: https://neurohive.io/en/popular-networks/vgg16/
7. Vinyals O. Show and Tell: A Neural Image Caption Generator [Electronic resourse]. Mode of access: https://arxiv.org/pdf/1411.4555.pdf
8. Dai B. Detecting Visual Relationships with Deep Relational Networks [Electronic resourse]. Mode of access: https://arxiv.org/pdf/1704.03114.pdf
9. Sadeghi M. Recognition Using Visual Phrases [Electronic resourse]. Mode of access: http://vision.cs.uiuc.edu/phrasal/recognition_using_visual_phrases.pdf
10. Lu C. Visual Relationship Detection with Language Priors [Electronic resourse]. Mode of access: https://arxiv.org/pdf/1608.00187.pdf
11. Krishna R. Visual Genome Connecting Language and Vision Using Crowdsourced Dense Image Annotations [Electronic resourse]. Mode of access: https://arxiv.org/pdf/1602.07332.pdf
12. Visual Genome [Electronic resourse]. Mode of access: https://visualgenome.org
13. Data Loading and Processing Tutorial [Electronic resourse]. Mode of access: https://pytorch.org/tutorials/beginner/data_loading_tutorial.html
14. TorchVision Models [Electronic resourse]. Mode of access: https://pytorch.org/docs/stable/torchvision/models.html
15. Ren S. Faster R-CNN: Towards Real-Time Object Detectionwith Region Proposal Networks [Electronic resourse]. Mode of access: https://papers.nips.cc/paper/5638-faster-r-cnn-towards-real-time-object-detection-with-region-proposal-networks.pdf
16. Chilamkurthy S. Transfer Learning Tutorial [Electronic resourse]. Mode of access: https://pytorch.org/tutorials/beginner/transfer_learning_tutorial.html

Одержано 10.03.2020

Про авторів:

Андон Пилип Іларіонович,
академік НАН України,
директор Інституту програмних систем НАН України.
Кількість наукових публікацій в українських виданнях – 400.
Кількість наукових публікацій в зарубіжних індексованих виданнях – 10.
http://orcid.org/0000-0001-6546-0826,

Глибовець Андрій Миколайович,
доктор технічних наук,
декан факультету інформатики
Національного університету «Києво-Могилянська академія».
Кількість наукових публікацій в українських виданнях – 40.
Кількість наукових публікацій в зарубіжних індексованих виданнях – 5.
https://orcid.org/0000-0003-4282-481X,

Куриляк Володимир Вікторович,
магістр факультету інформатики
Національного університету «Києво-Могилянська академія».

Місце роботи авторів:

Інститут програмних систем НАН України,
03187, м. Київ, проспект Академіка Глушкова, 40.

Національний університет «Києво-Могилянська академія»,
04070, м. Київ, вул. Сковороди 2.
E-mail: a.glybovets@ukma.edu.ua