СТВОРЕННЯ ВЕБ-ЗАСТОСУНКУ ТА АНАЛІЗ РЕЗУЛЬТАТІВ МОДЕЛІ

1.1 Перенавчання моделі Inception V3

Сучасні моделі розпізнавання зображень мають мільйони параметрів. Навчання їх з нуля потребує багато помічених навчальних даних та великої обчислювальної потужності (сотні годин GPU або більше). Передача навчання (transfer learning) – це техніка, яка швидше виконує цю роботу, витягуючи частину моделі, яка вже натренована на схожих задачах розпізнавання великого датасету зображень і повторно використовує отриману матрицю вагів в новій моделі. У даній роботі повторно використовуються можливості вилучення ознак зображень модулю з архітектурою Inception V3 [26], навченого на ImageNet, і натренований новий класифікаційний шар зверху. Inception V3 – архітектура нейронної мережі (рис. 3.1) для класифікації зображень.

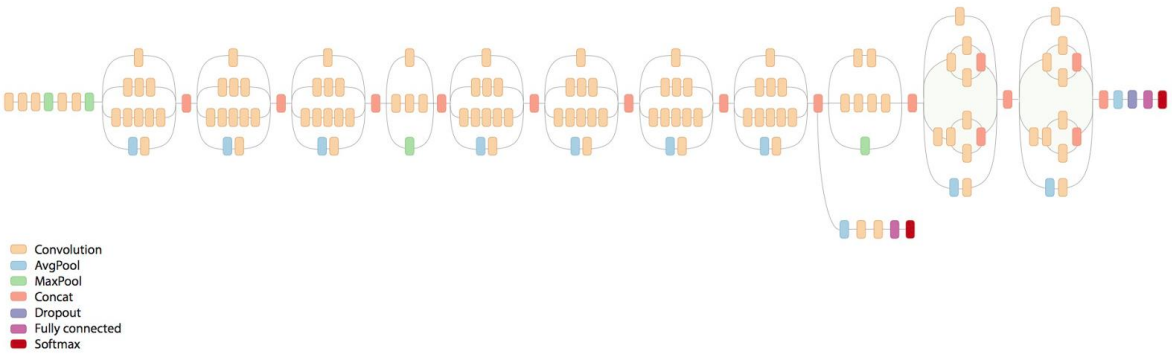

Рисунок 3.1 – Модель Inception V3 у комплекті з TensorFlow

Модуль TF-Hub використовує TF-Slim реалізацію цієї архітектури. Модуль містить навчений екземпляр мережі, підготований для отримання векторів ознак зображень. Контрольною точкою, експортованою в цей модуль,

була inception_v3_2016_08_28/inception_v3.ckpt, завантажена з попередньо підготовлених моделей TF-Slim. Її ваги спочатку були отримані шляхом навчання на наборі даних для класифікації зображень ImageNet. Цей модуль реалізує загальний підхід до обчислення векторів ознак зображення.

Перша фаза – аналіз всіх зображень на диску, обчислення і кешування значення «ущільнень» (bottleneck) для кожного з них. "Ущільнення" – неформальний термін, який часто використовують для позначення шару безпосередньо перед кінцевим вихідним шаром (вектором ознак зображення), який насправді займається класифікацією. Цей передостанній шар був навчений для виведення набору значень, які достатньо точні для того, щоб класифікатор використовував їх для розрізнення всіх класів, які треба було розпізнати. Це означає, що шар «ущільнення» повинен бути значущим та компактним описом зображень, оскільки він має містити достатньо інформації для класифікатора, щоб той зробив найбільш точний вибір у дуже малих наборах значень. Причина того, що перенавчання останнього шару може працювати на нових класах, полягає в тому, що вигляд інформації, необхідної для розрізнення всіх 1000 класів у ImageNet, часто також корисний для розрізнення нових видів об'єктів.

Після того, як створені шари «ущільнення», починається фактичне навчання останнього шару мережі. На цьому етапі виводиться серія кроків, кожна з яких показує точність тренування, точність перевірки та крос-ентропію. Точність тренування показує, який відсоток зображень, використаних у поточній навчальній множині, було позначено правильно. Точність перевірки – це точність на випадково вибраній групі зображень з тестової множини. Головна відмінність полягає в тому, що точність навчання залежить від зображень, яким мережа могла навчитися, тому мережа може «перенавчитися» (overfit) специфічним ознакам зображень (шуму) у

навчальних даних. Справжня міра точності мережі полягає в тому, щоб виміряти її точність на наборі даних, що не містився в навчальних даних, – це вимірюється за допомогою точності валідації (перевірки). Якщо точність на тренувальній вибірці висока, але точність валідації залишається низькою, це означає, що мережа «перенавчена» та запам'ятовує деякі особливості зображень з навчальної мноєини, які не є корисними в цілому. Крос-ентропія – це функція втрат, яка дає уявлення про те, наскільки успішно розвивається навчальний процес. Метою навчання є зменшення втрат настільки, наскільки це можливо. Оцінка крос-ентропії протягом всього процесу навчання дозволяє побачити, чи правильно працює нейронна мережа, незважаючи на короткочасний шум.

Навчання моделі було проведено за 8000 навчальних кроків. Кожен крок обирає 50 зображень навмання з навчальної множини, знаходить «ущільнення» з кеш-пам'яті та подає їх у фінальний шар, щоб отримати прогнози. Потім ці прогнози порівнюються з фактичними мітками, щоб оновити ваги останнього шару за допомогою процесу зворотнього поширення (back-propagation). З продовженням процесу точність повинна зростати, а крос-ентропія, у свою чергу, – спадати.

Хоча результати не настільки точні, як результати навчання повної моделі, це напрочуд ефективно для багатьох випадків. Даний підхід працює з невеликою кількістю навчальних даних, тобто, коли дано тисячі, а не мільйони позначених зображень [27].

Для розв’язання даної задачі було завантажено датасет творів образотворчого мистецтва з платформи для змагань з прогнозування моделей та аналітики, у рамках якої статисти та спеціалісти по роботі з даними конкурують з метою створення найкращих моделей для прогнозування та опису даних, завантажених компаніями та користувачами – Kaggle, а саме

змагання Painter By Number [28], основна задача якого – зрозуміти по двом зображенням, чи це картини одного і того ж художника – не перетинається з поставленою задачею цієї дипломної роботи – розпізнати стиль картини. Більшість зображень у цьому наборі даних було отримано з енциклопедії візуального мистецтва WikiArt. Після попередньої обробки: видалення пошкоджених, не розмічених, помилкових зображень, датасет складається з 74 070 картин, належних до 42 стилів образотворчого мистецтва.

Серед усіх стилів ОМ у даній задачі використовуються найбільш часто зображувані, тобто, кількість картин написаних у цьому стилі починається з 210 екземплярів. У табл. 3.1 наведено всі 42 стилі та кількість екземплярів стилю відповідно.

Таблиця 3.1 – Використані стилі образотворчого мистецтва

Abstract Art	759	Naturalism	377
Abstract Expressionism	1541	Naive Art (Primitivism)	1776
Academicism	766	Neo-Expressionism	337
Art Deco	517	Neo-Romanticism	464
Art Informel	987	Neoclassicism	1621
Art Nouveau (Modern)	3777	Op Art	412
Baroque	3252	Orientalism	294
Color Field Painting	675	Pointillism	387
Concretism	455	Pop Art	619
Constructivism	210	Post-Impressionism	4527
Cubism	1465	Realism	9895
Divisionism	259	Regionalism	249
Expressionism	5320	Renaissance	5147
Fauvism	564	Rococo	2101
Futurism	361	Romanticism	7037
Hard Edge Painting	293	Shin-hanga	316
Impressionism	8219	Surrealism	3131
Ink and wash painting	445	Symbolism	2625

Luminism	283	Sosaku hanga	329
Lyrical Abstraction	486	Tachisme	341
Minimalism	314	Ukiyo-e	1137

У результаті використання модулю TF-Hub та перенавчання останнього шару згорткової нейронної мережі, процес якого відбувався 25 годин, на виході було отримано модель (рис. 3.2) для розв’язку задачі розпізнавання стилю картин. У додатку А наведено логи процесу тренування, а саме тренувальна точність, крос-ентропія та валідацйна точність протягом 8000 кроків з інтервалом 50 кроків.

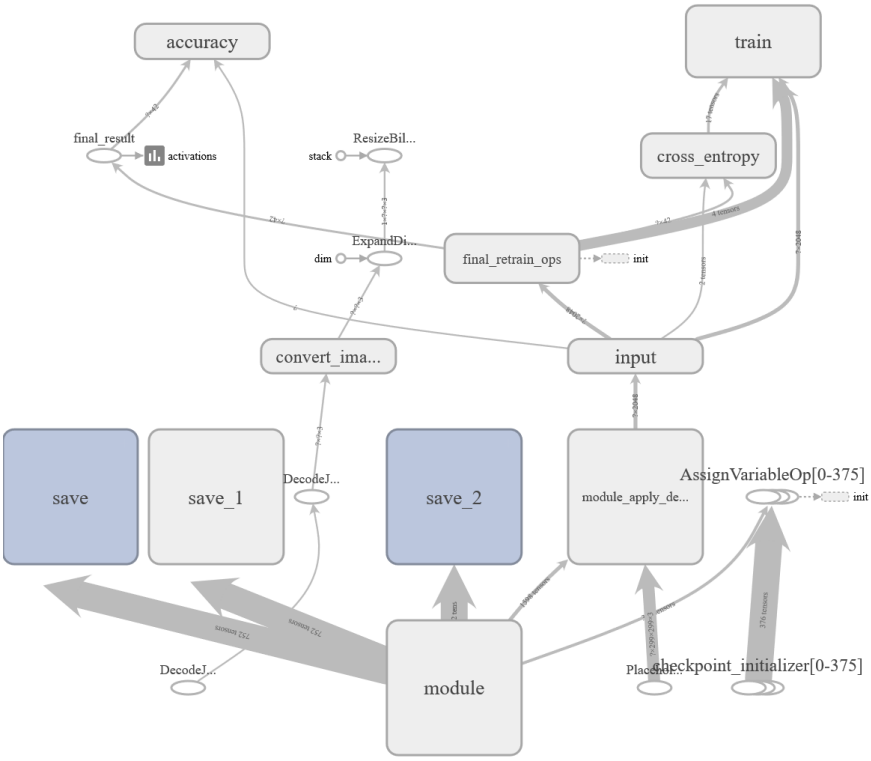

Рисунок 3.2 – Граф отриманої моделі розпізнавання стилю картин

За зміною основних чисельних показників моделі у ході навчання, наприклад, вагів моделі (рис. 3.3), можна спостерігати на згенерованих TensorBoard графіках та гістограмах.

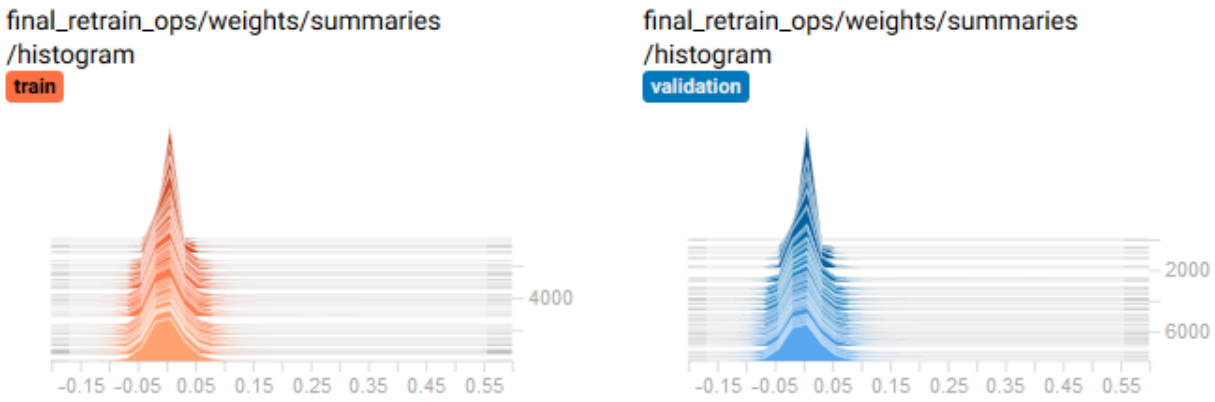

Рисунок 3.3 – Коригування вагів моделі протягом навчання

За результатами навчання згорткової нейронної мережі було отримано точність моделі 51,4%. Максимально досягнуте значення – 57,6%. На рис. 3.4 червоним відображена точність на навчальній вибірці, синім – на тестовій.

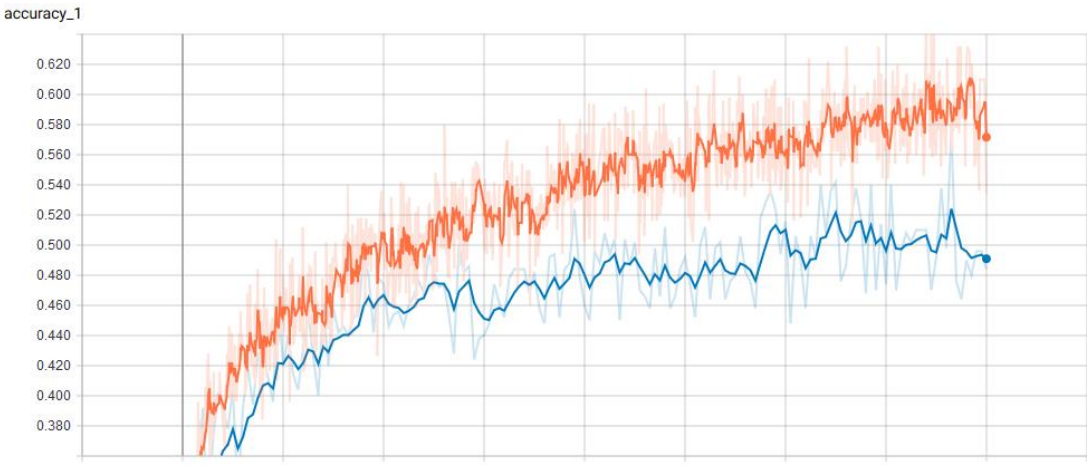

Рисунок 3.4 – Точність моделі протягом навчання

Так як розв’язується мультикласова задача, то невисока точність була очікувана. Для підвищення валідаційної точності моделі можна провести дефрагментацію (distortion) вихідних зображень, тим самим збільшуючи кількість вхідних даних для нейронної мережі та даючи їй більше можливостей для виокремлення необхідних абстрактних ознак зображення. Однак це потребує залучення комп’ютерних систем або дистанційних серверів

з набагато більшими обчислювальними потужностями, чим були використані у даній роботі.

3.2 Створення веб-застосунку

Для наочної презентації отриманої моделі та можливості її використання звичайними користувачами було створено веб-застосунок на Spring Boot, мова програмування Java. Spring Framework – це програмний каркас з підтримкою інверсії управління (конфігурація компонентів додатків і управління життєвим циклом об'єктів) для платформи Java. Проект зібраний за допомогою Maven – фреймворка для автоматизації роботи з програмними проектами на основі опису їх структури в файлах мовою POM. Графічне представлення продукту реалізоване засобами HTML&CSS.

TensorFlow надає ППІ (API) – Прикладний Програмний Інтерфейс (Application Programming Interface) – для використання в програмах Java. Ці ППІ підходять для завантаження моделей, створених у Python, та їх виконання у програмі Java. Інтеграцію моделі в програмний продукт забезпечила бібліотека libtensorflow-1.8.0.jar. Лістинг програмного продукту наведений у додатку Б.

Після запуску програми Apache Tomcat/8.5.14 починає роботу на порті 8080, тож за посиланням http://localhost:8080/ користувач бачить початкову сторінку веб-застосунку (рис. 3.5).

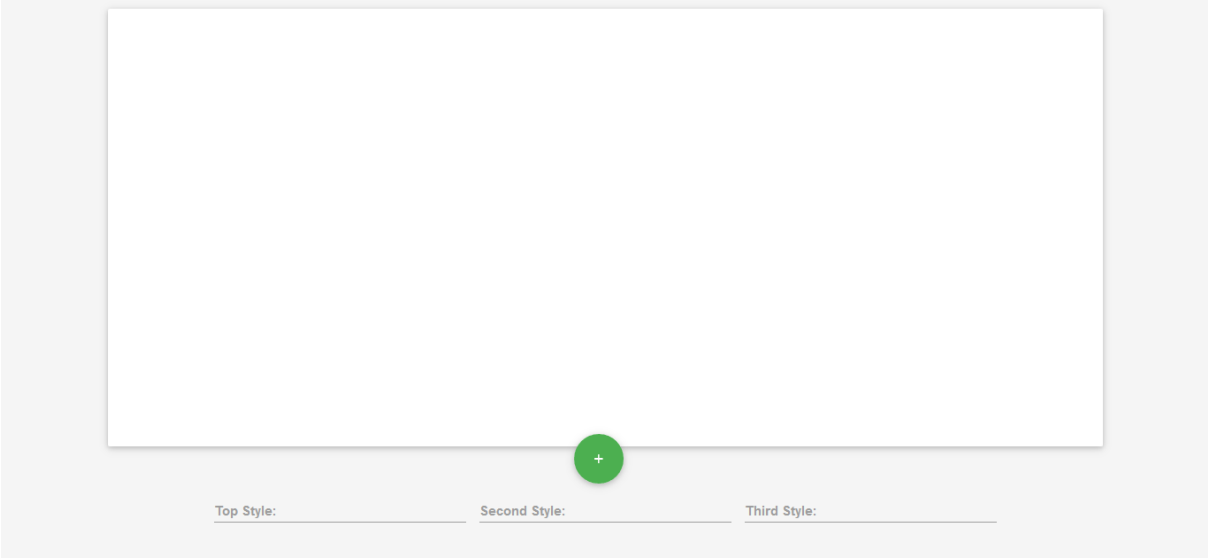

Рисунок 3.5 – Початкова сторінка веб-застосунку

Інтерфейс додатку простий та зрозумілий у використанні. Натискаючи на кнопку «+», користувач може завантажити зображення картини, стиль якої він хоче дізнатися. Після цього програма обробляє це зображення, попередньо готуючи його до використання моделлю: обтинає до меншого розміру (299×299) для швидкодії застосунку та перетворює його на тензор розмірності [1, 299, 299, 3]. На виході моделі програма отримує тензор ймовірностей належності до класів розмірністю [1, 42] відповідно кількості стилів. Далі відбувається обробка результатів, а саме виокремлення трьох стилів з найбільшими ймовірностями належності. Результат – стиль та відсоток, з яким картину можна до нього віднести – виводиться у відповідні лейбли під основну форму зображення на сторінці веб-застосунку.

Наприклад, після завантаження «Spider Of The Evening» (1990) Сальвадора Далі, яка, як відомо, належить до стилю сюрреалізму, можна побачити розподіл стилів моделлю (рис. 3.6).

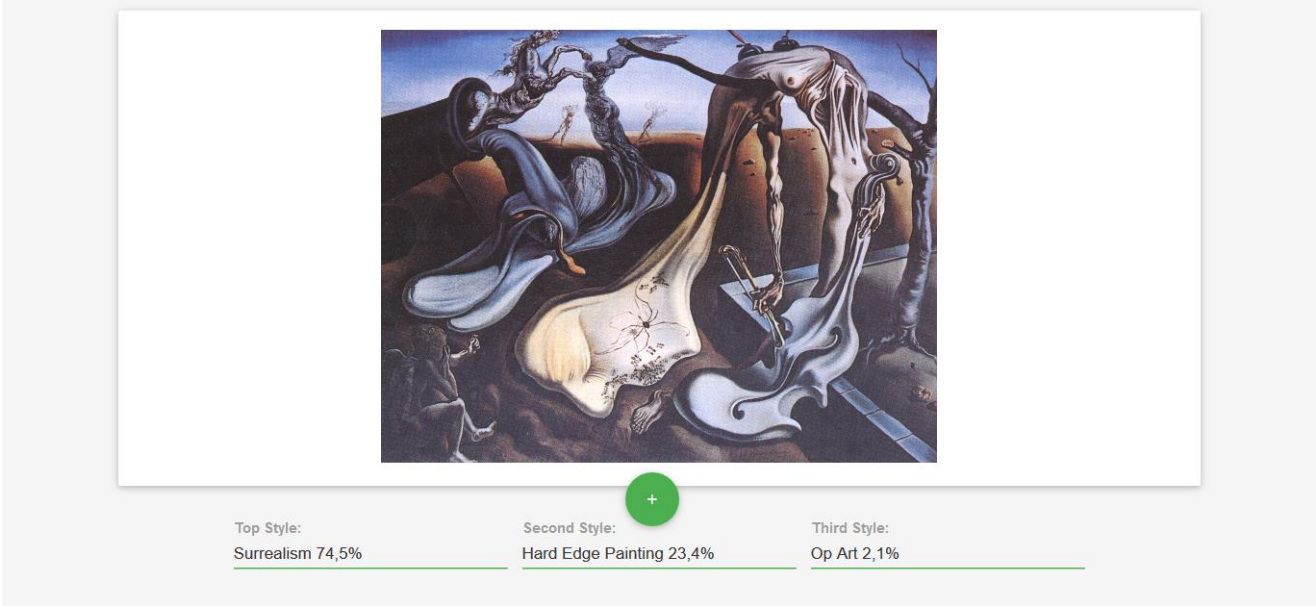

Рисунок 3.6 – Розпізнавання сюрреалізму на прикладі «Spider Of The Evening» Сальвадора Далі

Як можна бачити з результатів, сюрреалізм був розпізнаний успішно – він займає перше місце по ймовірності належності. Успішне розпізнавання можна також продемонструвати на зовсім іншому по абстрактним ознакам стилю – пуантилізму – на прикладі картини Анрі Едмона Кросса «У тіні» (1902) (рис. 3.7).

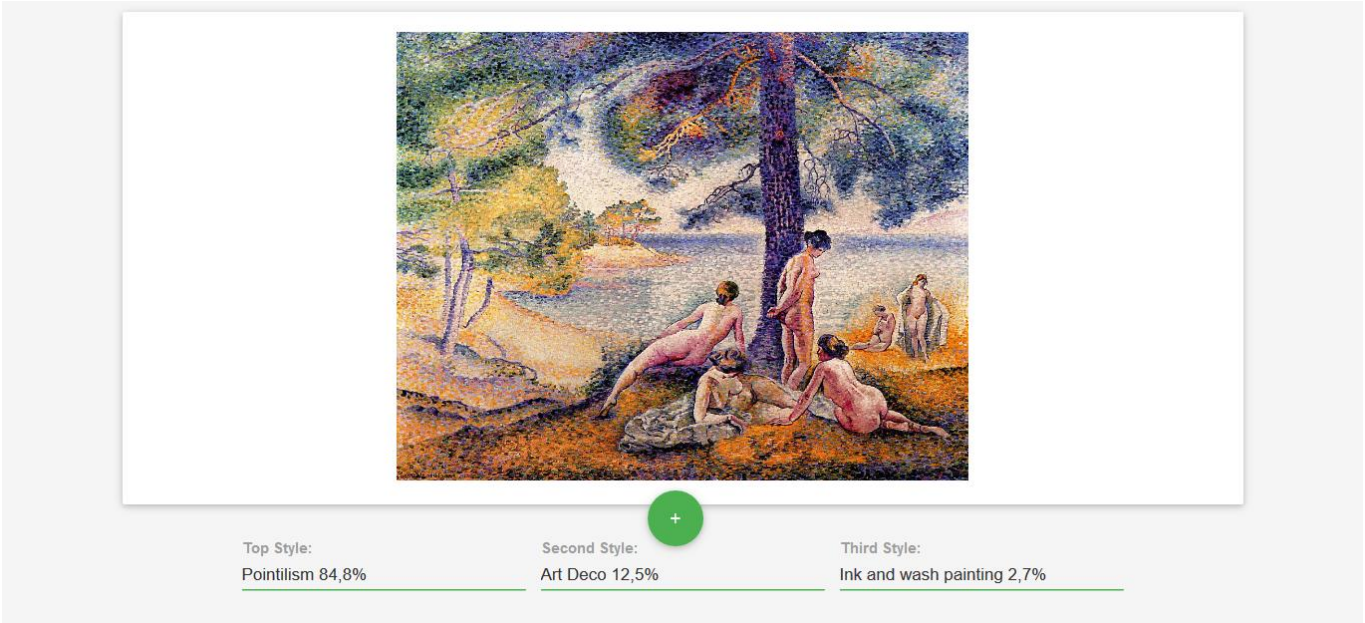

Рисунок 3.7 – Розпізнавання пуантилізму на прикладі «У тіні»
Анрі Едмона Кросса

Відповідно до точності натренованої моделі, застосунок розпізнає стилі неправильно доволі часто. Таким прикладом є «Зоряна ніч» (1889) Вінсента Ван Гога (рис. 3.8).

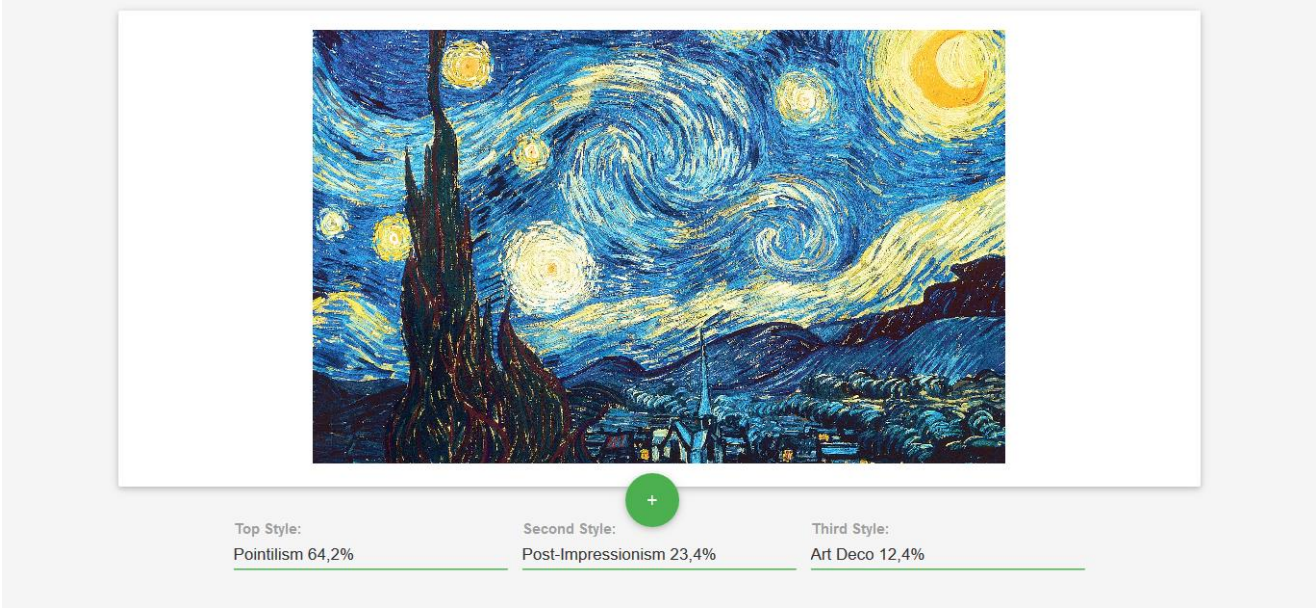

Рисунок 3.8 – Розпізнавання постімпресіонізму на прикладі «Зоряна ніч»
Вінсента Ван Гога

Згідно з результатами найбільша ймовірність належності до пуантилізму, хоча правильний стиль – постімпресіонізм – посідає друге місце. Навіть порівнявши дві останні картини, можна побачити, що вони доволі схожі в манері нанесення фарби та палітрою кольорів. Якщо нейронна мережа виділила ці абстрактні ознаки для розподілення між класами, як основні, то це пояснює такий розподіл ймовірностей. Все ж таки, наявність коректного стилю в топ-3 означає, що основні вектори ознак натренованої моделі сформовані правильно. Додаткові детальніші абстрактні ознаки можна отримати з залученням більших обчислювальних потужностей, як було описано вище.