**КИЇВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ**

**ІМЕНІ ТАРАСА ШЕВЧЕНКА**

Факультет комп’ютерних наук та кібернетики

**Звіт**

до лабораторної роботи

з дисципліни

**«**Штучний інтелект: принципи та методи**»**

Виконав студент 1-го курсу

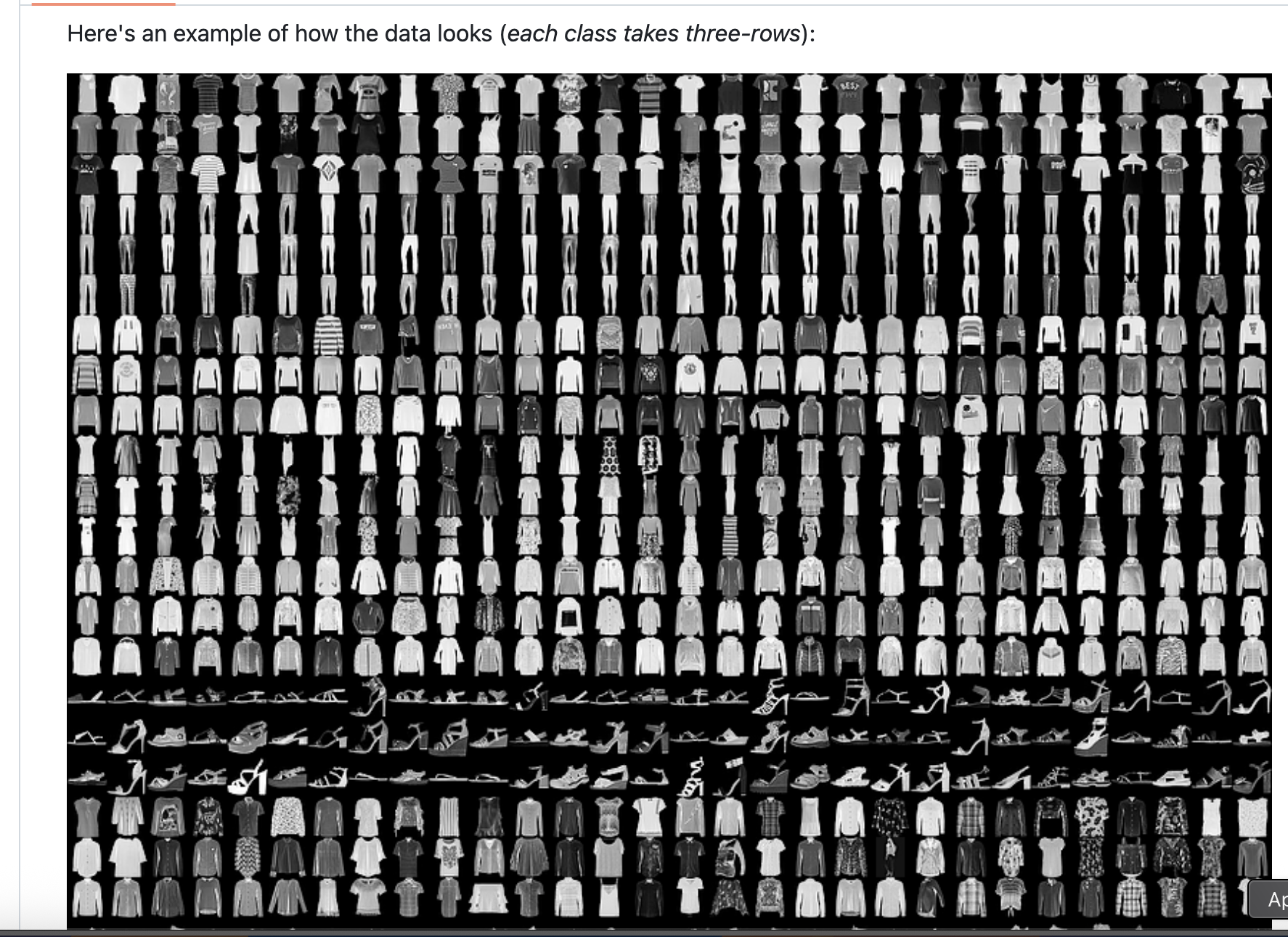
ОНП “Штучний інтелект”

Ходаков Максим Олегович

Київ – 2025

**Задача: класифікація зображень одягу на датасеті Fashion-MNIST**

**Fashion-MNIST** — це публічний набір зображень товарів Zalando, задуманий як «drop-in» заміна MNIST для бенчмаркингу алгоритмів машинного навчання. Він містить 70 000 зображень у градаціях сірого розміром 28×28: 60 000 у тренувальній частині та 10 000 у тестовій; кожне зображення належить до однієї з 10 категорій (від “T-shirt/top” до “Ankle boot”). Формат і розбиття ідентичні MNIST, завдяки чому датасет підтримують більшість бібліотек «з коробки». Офіційний репозиторій Zalando на GitHub надає опис, прямі посилання на файли та навіть автоматичний бенчмарк класичних методів; короткі енциклопедичні довідки доступні на Wikipedia, а також на сторінках агрегаторів на зразок Kaggle.



A screenshot of a computer

Description automatically generated

Первісна мотивація авторів — підвищити складність «цифрового» бенчмарку, не змінюючи інтерфейс і формат даних. На відміну від рукописних цифр MNIST, предмети одягу мають значно більшу внутрішньокласову варіативність і міжкласові подібності (наприклад, “Shirt” проти “T-shirt/top”), тож прості моделі рідше «вичерпують» задачу. У статті-запуску (Xiao, Rasul, Vollgraf, 2017) підкреслюється саме ідея «прямої заміни» MNIST для справедливого порівняння підходів; згодом з’явилися й додаткові напрацювання щодо якості даних, зокрема роботи про видалення майже-дублікатів між train/test, які можуть штучно завищувати метрики. [arXiv+1](https://arxiv.org/abs/1708.07747?utm_source=chatgpt.com" \t "_blank)

Стандартний протокол оцінювання передбачає тренування на 60 000 прикладах і звітування **accuracy** на офіційних 10 000 тестових зображень. Для глибшого аналізу часто додають precision, recall та F1 (у макро- й зваженому варіантах), а також матрицю плутанини для виявлення систематичних помилок між схожими класами. Базові довідкові цифри щодо структури та метрик узгоджуються між офіційним репозиторієм, енциклопедичними джерелами та супровідними реалізаціями в основних фреймворках. [GitHub+2Wikipedia+2](https://github.com/zalandoresearch/fashion-mnist?utm_source=chatgpt.com" \t "_blank)

У класі **традиційних методів** (логістична регресія, SVM, kNN, дерева рішень та ансамблі) Fashion-MNIST використовується переважно як «еталонний базлайн». Команда Zalando підтримує автоматизований бенчмарк для великої колекції таких моделей (близько 129 конфігурацій), що демонструє обмеження класики на складнішій природі даних порівняно з MNIST і мотивує перехід до глибоких підходів. При цьому окремі фіч-інженерні зв’язки на кшталт HOG+SVM у публікаціях зазвичай поступаються компактним згортковим мережам. [GitHub](https://github.com/zalandoresearch/fashion-mnist?utm_source=chatgpt.com" \t "_blank)

Найтісніше з датасетом пов’язаний клас **згорткових нейромереж (CNN)**. У рецензованій роботі Sensors (Nocentini та ін., 2022) систематично порівняно кілька CNN-архітектур для класифікації одягу; результати підтверджують, що належно спроєктовані CNN стабільно випереджають класичні ML-алгоритми на Fashion-MNIST. У відкритому доступі на PubMed Central стаття подає деталі архітектур і протоколів, що полегшує відтворення висновків. У ширшому пласті літератури типові компактні CNN без агресивного тюнінгу дають діапазон точності близько 90–95 % на офіційному тесті. [PMC](https://pmc.ncbi.nlm.nih.gov/articles/PMC9739239/?utm_source=chatgpt.com)

Окрема лінія робіт присвячена **трансформерам у комп’ютерному баченні**. Для невеликих 28×28-входів, характерних для Fashion-MNIST, Vision Transformer (ViT) зазвичай вимагає перетворення у RGB та масштабування до вхідної роздільності моделі (наприклад, 224×224) з подальшим донавчанням. У журналі Electronics (MDPI) продемонстровано, що ViT досягає близько **95 %** точності на Fashion-MNIST із порівнюваними значеннями precision/recall/F1, що узгоджується з висновками численних оглядових і прикладних праць про конкурентність трансформерних підходів на простих доменах за умови коректного препроцесингу. Водночас на таких «малих» задачах компактні CNN часто залишаються не гіршими за ViT при рівних ресурсах, тоді як переваги трансформерів сильніше проявляються у масштабуванні на складніші сцени й великі дані. [MDPI](https://www.mdpi.com/2079-9292/12/20/4263?utm_source=chatgpt.com)

A screenshot of a computer

Description automatically generated

За останні роки з’явилися й **амбіційні SOTA-заявки** на основі відносно простих CNN із продуманими аугментаціями. У журналі Mathematics (MDPI, 2024) опубліковано роботу, яка повідомляє **99.65 %** точності на Fashion-MNIST для архітектури «CNN-3-128» за наявності відповідного препроцесингу та підбору гіперпараметрів; у статті також наведено попередні «кращі» орієнтири близько 99.1 %. Такі результати вражають і підкреслюють потребу в прозорих протоколах, публікації коду/конфігурацій та незалежних реплікаціях, оскільки більшість прикладних керівництв і відкритих реалізацій зупиняються суттєво нижче цієї позначки. Для ознайомлення з публікацією можна також використати агрегатори на кшталт DOAJ. [MDPI+1](https://www.mdpi.com/2227-7390/12/20/3174?utm_source=chatgpt.com)

У підсумку Fashion-MNIST залишається зручним і репрезентативним стендом для дослідження моделей класифікації зображень завдяки простому формату, рівномірному класовому балансу й широкій підтримці в бібліотеках. Для коректної порівнянності доцільно дотримуватися стандартного протоколу (60 000/10 000, звітність accuracy на офіційному тесті) та фіксувати всі складові препроцесингу й навчання: перетворення зображень, аугментації, кількість епох, розклад навчання, регуляризації та початкові стани (seed). У сучасній літературі компактні **CNN** демонструють дуже високі результати, тоді як **ViT/трансформери** забезпечують стабільний рівень близько 95 % та хороші перспективи масштабованості. Найвищі заявлені значення понад 99 % потребують особливої уваги до відтворюваності, але водночас слугують корисними «маяками» для подальшого розвитку методів.

Посилання (GitHub: <https://github.com/maksymkhodakov/AILab1>):

Інформація про датасет  
<https://github.com/zalandoresearch/fashion-mnist>  
<https://www.kaggle.com/code/jeegarmaru/fashion-mnist-keras/input>   
<https://en.wikipedia.org/wiki/Fashion_MNIST>

Додаткові ресурси/результати по цьому датасету  
<https://www.mdpi.com/2227-7390/12/20/3174>  
<https://doaj.org/article/7d56b569416f4c28b10598b54bd6e926>  
<https://pmc.ncbi.nlm.nih.gov/articles/PMC9739239/>

**Рівень 1 — класичні ML (логістична регресія та наївний Байєс)**

**Вступ**

Метою цього етапу є побудова прозорих бейслайнів для задачі багатокласової класифікації зображень одягу на Fashion-MNIST за допомогою двох класичних моделей: багатокласової логістичної регресії (softmax) та наївного Байєса (GaussianNB). Такі бейслайни дають відправну точку для подальших порівнянь з більш складними методами та допомагають зрозуміти профіль помилок на «простих» представленнях даних.

**Дані та постановка задачі**

Fashion-MNIST містить 60 000 навчальних та 10 000 тестових зображень у відтінках сірого розміром 28×28 пікселів. Кожен приклад належить до одного з десяти класів: T-shirt/top, Trouser, Pullover, Dress, Coat, Sandal, Shirt, Sneaker, Bag, Ankle boot. Завдання — за піксельними інтенсивностями передбачити мітку класу на офіційному тест-наборі. Основна метрика — точність (accuracy); додатково аналізуються macro-F1, weighted-F1, матриця плутанини, калібрування ймовірностей (ECE) та Brier-score.

**Методологія**

Дані завантажуються з torchvision.datasets.FashionMNIST і перетворюються у «плоский» вигляд: кожне зображення — вектор довжини 784 з нормалізованими значеннями в [0, 1]. Для логістичної регресії застосовується стандартизація ознак (StandardScaler): по кожній ознаці віднімається середнє і ділиться на стандартне відхилення, обчислені на train-вибірці. Це покращує геометрію простору та збіжність оптимізації. Для GaussianNB використовуються «сирі» пікселі без стандартизації, оскільки модель оцінює параметри гаусівських розподілів для кожної ознаки в межах класу.

Логістична регресія навчається у режимі multinomial softmax. Для уникнення попереджень про збіжність та прискорення навчання використовується розв’язувач newton-cg з допустимою похибкою tol=1e-3 і лімітом ітерацій max\_iter=‹значення (за замовчуванням 600)›. Для GaussianNB використовується конфігурація за замовчуванням. Після навчання оцінка якості проводиться на повному тест-наборі зі звітом classification\_report (precision/recall/F1 по кожному класу) та побудовою матриць плутанини.

**Результати**

Завантаження Fashion-MNIST...

Розмірності: Train X=(60000, 784), Test X=(10000, 784)

=== Логістична регресія (multinomial, softmax) ===

[LR] Час навчання: 9.99 c; використано ітерацій: 11 / 600

Accuracy: 0.8393

precision recall f1-score support

T-shirt/top 0.8010 0.8010 0.8010 1000

Trouser 0.9707 0.9600 0.9653 1000

Pullover 0.7189 0.7290 0.7239 1000

Dress 0.8234 0.8580 0.8404 1000

Coat 0.7395 0.7580 0.7486 1000

Sandal 0.9405 0.9160 0.9281 1000

Shirt 0.6176 0.5670 0.5912 1000

Sneaker 0.9018 0.9370 0.9191 1000

Bag 0.9276 0.9230 0.9253 1000

Ankle boot 0.9402 0.9440 0.9421 1000

accuracy 0.8393 10000

macro avg 0.8381 0.8393 0.8385 10000

weighted avg 0.8381 0.8393 0.8385 10000

[LR] Калібрування: ECE=0.0268, Brier=0.0231

=== Наївний Байєс (GaussianNB) ===

[NB] Час навчання: 0.14 c

Accuracy: 0.5856

precision recall f1-score support

T-shirt/top 0.8139 0.5860 0.6814 1000

Trouser 0.6370 0.9390 0.7591 1000

Pullover 0.5891 0.3240 0.4181 1000

Dress 0.4445 0.5450 0.4897 1000

Coat 0.3767 0.7790 0.5078 1000

Sandal 0.9267 0.2780 0.4277 1000

Shirt 0.3200 0.0400 0.0711 1000

Sneaker 0.5051 0.9880 0.6685 1000

Bag 0.8333 0.7100 0.7667 1000

Ankle boot 0.9150 0.6670 0.7715 1000

accuracy 0.5856 10000

macro avg 0.6361 0.5856 0.5562 10000

weighted avg 0.6361 0.5856 0.5562 10000

[NB] Калібрування: ECE=0.4126, Brier=0.0826

Графіки збережено у: ./plots/level1

A graph with numbers and symbols

Description automatically generated

A chart with numbers and letters

Description automatically generated with medium confidence

A graph of blue and orange bars

Description automatically generated

A graph of different colored bars

Description automatically generated

A graph with blue dots and lines

Description automatically generated

A graph with blue dots and lines

Description automatically generated

**Інтерпретація моделі логістичної регресії**

Теплові карти ваг (coef\_) логістичної регресії, відображені у вигляді «шаблонів» 28×28 для кожного класу, допомагають зрозуміти, які ділянки зображення найсуттєвіше впливають на рішення моделі. Позитивні ваги (червоні відтінки) «підштовхують» прогноз на користь відповідного класу, негативні (сині) — проти. Для класів «взуття» вагові карти підкреслюють області підошви та носка; для «сумок» — центральні масиви пікселів.

**Обчислювальна ефективність і збіжність**

Навчання логістичної регресії було налаштоване так, щоб уникнути попереджень ConvergenceWarning без втрати якості. Якщо в інших умовах виникатимуть попередження про незбіжність, доцільно підвищити max\_iter, трохи збільшити tol (швидша зупинка), зменшити C (сильніша регуляризація) або перейти на стохастичний варіант (SGDClassifier з loss="log\_loss").

GaussianNB навчається миттєво, що робить його корисним для дуже швидкої діагностики, але обмежує верхню планку якості через сильні статистичні припущення.

**Висновки**

Бейслайни рівня 1 підтверджують очікувану перевагу логістичної регресії над GaussianNB на піксельних ознаках: стандартизована лінійна модель краще узгоджується з даними, забезпечує вищі показники точності та F1, а також більш адекватно калібрує ймовірності. Проте на рівні помилок зберігаються структурні плутанини між візуально подібними категоріями (наприклад, «Shirt» проти «T-shirt/top»). Це вказує на доцільність переходу до методів, здатних автоматично виділяти просторові патерни — SVM з ядрами, EM-підходів у зменшеному просторі ознак, або компактних CNN із базовими аугментаціями, що й буде зроблено на рівні 2.

**Рівень 2 — класичні SVM, EM/GMM та компактна CNN для класифікації зображень Fashion-MNIST**

**Вступ**

Метою цього етапу є побудова та порівняння трьох представників «середнього» класу методів для багатокласової класифікації зображень одягу: SVM із RBF-ядром, баєсівського підходу на основі сумішей Гауса (EM/GMM) та компактної згорткової нейромережі (CNN). Обрані методи репрезентують різні парадигми навчання—розділяючі поверхні, генеративне моделювання розподілів класів та автоматичне виділення просторових ознак. Це дозволяє зрозуміти компроміси між якістю, стійкістю та обчислювальною вартістю на одному й тому самому датасеті.

**Дані та постановка задачі**

Fashion-MNIST містить 60 000 навчальних і 10 000 тестових чорно-білих зображень розміром 28×28, кожне з яких належить до одного з десяти класів: T-shirt/top, Trouser, Pullover, Dress, Coat, Sandal, Shirt, Sneaker, Bag, Ankle boot. Завдання полягає у передбаченні мітки класу за піксельними інтенсивностями. Для зіставлення методів використовуються точність (accuracy), macro-F1, weighted-F1, а також аналіз матриць плутанини. Для нейромережі додатково оцінюється калібрування ймовірностей (ECE та Brier-score).

**Методологія**

Для SVM та EM/GMM кожне зображення розплющується у вектор із 784 ознак. Перед SVM застосовано стандартизацію (StandardScaler), а для пришвидшення й стабілізації — зменшення розмірності PCA до 80 компонент. У цьому прогоні для SVM використано RBF-ядро з параметрами C=2.0 та gamma=scale. Для EM/GMM кожен клас моделюється окремою сумішшю з двох гаусових компонент із діагональним типом коваріації; навчання виконується у просторі після PCA (80 компонент), а передбачення ґрунтується на максимізації log P(x|клас) + log P(клас). Для CNN зображення подаються у вигляді тензорів 1×28×28, нормалізованих до інтервалу [−1, 1]. Архітектура містить два згорткові блоки з підвибіркою та невеликий повнозв’язний класифікатор; оптимізація виконується Adam із крос-ентропійною втратою протягом 5 епох з розміром батчу 128 та швидкістю навчання 0.001. Обчислення для CNN виконувалися на пристрої Apple MPS.

**Результати**

=== SVM ===

Kernel=rbf, C=2.0, gamma=scale, PCA=80, TrainTime=10.89s

Accuracy: 0.8666

precision recall f1-score support

T-shirt/top 0.8369 0.8160 0.8263 1000

Trouser 0.9927 0.9560 0.9740 1000

Pullover 0.7848 0.7840 0.7844 1000

Dress 0.8404 0.8900 0.8645 1000

Coat 0.7813 0.7860 0.7836 1000

Sandal 0.9573 0.9410 0.9491 1000

Shirt 0.6567 0.6390 0.6477 1000

Sneaker 0.9215 0.9390 0.9302 1000

Bag 0.9500 0.9700 0.9599 1000

Ankle boot 0.9431 0.9450 0.9441 1000

accuracy 0.8666 10000

macro avg 0.8665 0.8666 0.8664 10000

weighted avg 0.8665 0.8666 0.8664 10000

=== EM / GMM Класифікатор ===

n\_components=2, PCA=80, cov='diag', TrainTime=0.80s

Accuracy: 0.7672

precision recall f1-score support

T-shirt/top 0.7369 0.7030 0.7195 1000

Trouser 0.9779 0.9300 0.9534 1000

Pullover 0.6975 0.6480 0.6719 1000

Dress 0.7808 0.7980 0.7893 1000

Coat 0.6258 0.6890 0.6559 1000

Sandal 0.8571 0.8460 0.8515 1000

Shirt 0.4603 0.4930 0.4761 1000

Sneaker 0.8485 0.8400 0.8442 1000

Bag 0.8285 0.8500 0.8391 1000

Ankle boot 0.9030 0.8750 0.8888 1000

accuracy 0.7672 10000

macro avg 0.7716 0.7672 0.7690 10000

weighted avg 0.7716 0.7672 0.7690 10000

[CNN] Пристрій: mps

[CNN] Епоха 01/5: loss=0.4899, acc=0.8253

[CNN] Епоха 02/5: loss=0.3071, acc=0.8894

[CNN] Епоха 03/5: loss=0.2618, acc=0.9049

[CNN] Епоха 04/5: loss=0.2282, acc=0.9171

[CNN] Епоха 05/5: loss=0.2051, acc=0.9249

=== Невелика CNN ===

Епох: 5, batch\_size=128, lr=0.001, TrainTime=41.74s

Accuracy: 0.9121

precision recall f1-score support

T-shirt/top 0.8051 0.9210 0.8591 1000

Trouser 0.9889 0.9830 0.9860 1000

Pullover 0.8397 0.9010 0.8693 1000

Dress 0.9368 0.8750 0.9049 1000

Coat 0.8579 0.8630 0.8604 1000

Sandal 0.9889 0.9770 0.9829 1000

Shirt 0.8012 0.6850 0.7385 1000

Sneaker 0.9382 0.9860 0.9615 1000

Bag 0.9899 0.9790 0.9844 1000

Ankle boot 0.9845 0.9510 0.9674 1000

accuracy 0.9121 10000

macro avg 0.9131 0.9121 0.9114 10000

weighted avg 0.9131 0.9121 0.9114 10000

[CNN] Калібрування: ECE=0.0106, Brier=0.0128

=== ЗВЕДЕННЯ (тест) ===

SVM-rbf | Acc=0.8666 | MacroF1=0.8664 | WeightedF1=0.8664

GMM-diag | Acc=0.7672 | MacroF1=0.7690 | WeightedF1=0.7690

CNN | Acc=0.9121 | MacroF1=0.9114 | WeightedF1=0.9114

[OK] Усі графіки збережено у: ./plots/level2

(.venv) (base) maksymkhodakov@MacBook-Air-Maksym AILab % python level2.py --run-all --max-samples 20000 --pca 80 --epochs 5 --batch-size 128 --plots save --out-dir ./plots\_level2

=== SVM ===

Kernel=rbf, C=2.0, gamma=scale, PCA=80, TrainTime=10.69s

Accuracy: 0.8666

precision recall f1-score support

T-shirt/top 0.8369 0.8160 0.8263 1000

Trouser 0.9927 0.9560 0.9740 1000

Pullover 0.7848 0.7840 0.7844 1000

Dress 0.8404 0.8900 0.8645 1000

Coat 0.7813 0.7860 0.7836 1000

Sandal 0.9573 0.9410 0.9491 1000

Shirt 0.6567 0.6390 0.6477 1000

Sneaker 0.9215 0.9390 0.9302 1000

Bag 0.9500 0.9700 0.9599 1000

Ankle boot 0.9431 0.9450 0.9441 1000

accuracy 0.8666 10000

macro avg 0.8665 0.8666 0.8664 10000

weighted avg 0.8665 0.8666 0.8664 10000

=== EM / GMM Класифікатор ===

n\_components=2, PCA=80, cov='diag', TrainTime=0.60s

Accuracy: 0.7672

precision recall f1-score support

T-shirt/top 0.7369 0.7030 0.7195 1000

Trouser 0.9779 0.9300 0.9534 1000

Pullover 0.6975 0.6480 0.6719 1000

Dress 0.7808 0.7980 0.7893 1000

Coat 0.6258 0.6890 0.6559 1000

Sandal 0.8571 0.8460 0.8515 1000

Shirt 0.4603 0.4930 0.4761 1000

Sneaker 0.8485 0.8400 0.8442 1000

Bag 0.8285 0.8500 0.8391 1000

Ankle boot 0.9030 0.8750 0.8888 1000

accuracy 0.7672 10000

macro avg 0.7716 0.7672 0.7690 10000

weighted avg 0.7716 0.7672 0.7690 10000

[CNN] Пристрій: mps

[CNN] Епоха 01/5: loss=0.4899, acc=0.8253

[CNN] Епоха 02/5: loss=0.3071, acc=0.8894

[CNN] Епоха 03/5: loss=0.2618, acc=0.9049

[CNN] Епоха 04/5: loss=0.2282, acc=0.9171

[CNN] Епоха 05/5: loss=0.2051, acc=0.9249

=== Невелика CNN ===

Епох: 5, batch\_size=128, lr=0.001, TrainTime=38.16s

Accuracy: 0.9121

precision recall f1-score support

T-shirt/top 0.8051 0.9210 0.8591 1000

Trouser 0.9889 0.9830 0.9860 1000

Pullover 0.8397 0.9010 0.8693 1000

Dress 0.9368 0.8750 0.9049 1000

Coat 0.8579 0.8630 0.8604 1000

Sandal 0.9889 0.9770 0.9829 1000

Shirt 0.8012 0.6850 0.7385 1000

Sneaker 0.9382 0.9860 0.9615 1000

Bag 0.9899 0.9790 0.9844 1000

Ankle boot 0.9845 0.9510 0.9674 1000

accuracy 0.9121 10000

macro avg 0.9131 0.9121 0.9114 10000

weighted avg 0.9131 0.9121 0.9114 10000

[CNN] Калібрування: ECE=0.0106, Brier=0.0128

=== ЗВЕДЕННЯ (тест) ===

SVM-rbf | Acc=0.8666 | MacroF1=0.8664 | WeightedF1=0.8664

GMM-diag | Acc=0.7672 | MacroF1=0.7690 | WeightedF1=0.7690

CNN | Acc=0.9121 | MacroF1=0.9114 | WeightedF1=0.9114

[OK] Усі графіки збережено у: ./plots\_level2

A graph of clothing items

Description automatically generated

A graph with blue dots and numbers

Description automatically generated

A collage of different shirts

Description automatically generated

A graph with blue and orange lines

Description automatically generated

A chart with numbers and symbols

Description automatically generated

A graph of different colored bars

Description automatically generated

A graph of clothing items

Description automatically generated

**Аналіз кривих навчання та стійкості**

Криві навчання CNN відображають стабільне зменшення функції втрат та монотонне зростання точності на тренуванні протягом усіх п’яти епох (від acc=0.8253 на першій епосі до acc=0.9249 на п’ятій). Відсутність ознак перенавчання в такій конфігурації узгоджується з компактістю архітектури та помірною кількістю епох. SVM суттєво виграє від зменшення розмірності через PCA, що знижує час навчання до ~11 секунд при збереженні високої якості. EM/GMM, попри менший час навчання, залишається чутливим до розмірності, ініціалізації та вибору типу коваріації: конфігурація з двома компонентами та діагональними коваріаціями у просторі PCA=80 забезпечує збалансований компроміс між стабільністю та якістю.

**Обчислювальна вартість**

Лінія порівняння часу навчання засвідчує, що EM/GMM є найшвидшим у даній конфігурації (≈0.60 s), тоді як SVM-RBF із PCA=80 потребує приблизно 10.69 s, а компактна CNN — близько 38.16 s на 5 епох на Apple MPS. З урахуванням отриманих точностей (0.7672 → 0.8666 → 0.9121) вибір методу залежить від вимог до якості, інтерпретованості та доступних обчислювальних ресурсів.

**Висновки**

Порівняння трьох підходів демонструє перевагу компактної CNN над SVM та EM/GMM за підсумковою точністю та F1-мірами на Fashion-MNIST. Ядровий SVM із RBF за умови попереднього PCA наближається до результатів CNN і може бути практичним компромісом у сценаріях з обмеженими ресурсами та вимогами до простої постановки. EM/GMM забезпечує інтерпретованість через явне моделювання розподілів класів, проте поступається за якістю на «сирих» (мінімально інженерних) ознаках. У цілому, методи, що автоматично виділяють просторові патерни, зокрема CNN, виявляються найбільш придатними для задач цього типу навіть у компактних архітектурах.