**1 АНАЛІЗ ТА ДОСЛІДЖЕННЯ СИСТЕМ РОЗПІЗНАВАННЯ РУКОПИСНОГО ТЕКСТУ**

**1.1 Актуальність проблеми**

За сучасних темпів розвитку та поширення обчислювальної техніки набуває актуальності широкий клас задач, пов'язаних із проблемою організації взаємодії людини й ЕОМ. Існуючі на сьогоднішній день технічні засоби введення інформації є оптимальним вирішенням даної проблеми з точки зору простоти реалізації, але вони не є природними для людини, отже, вимагають від неї спеціальної підготовки.

Комп‘ютери завжди існували для того, щоб допомагати людині, для цього необхідно передати інформацію до комп’ютера та назад і зрозуміло, що чим легше це зробити, тим більше від нього користі, найлегші шляхи – звичні, звичний шлях спілкування – натуральна мова, отже чим краще спроможний комп’ютер спілкуватися людськими мовами тим більше від нього користі незалежно від того в якій області він використовується.

Ефективний спосіб покращення взаємодії людини з ЕОМ - інтелектуалізація взаємодії за допомогою сучасних досягнень в області цифрової обробки інформації і штучного інтелекту. Одна з найбільш розповсюджених задач, пов'язаних із проблемою організації взаємодії людини й ЕОМ, - це розпізнавання рукописного тексту.

Розпізнавання рукописного введення- це здатність комп'ютера отримувати та інтерпретувати рукописне введення. Розпізнавання тексту може проводитися "оффлайновим" методом з уже написаного на папері тексту або "онлайновим" методом зчитуванням рухів кінчика ручки, приміром по поверхні спеціального комп'ютерного екрану.

Оптичне розпізнавання (OCR) - механічний або електронний переказ зображень рукописного, машинописного або друкованого тексту в текстові дані - послідовність кодів, що використовуються для представлення символів в комп'ютері(наприклад, в текстовому редакторі). Розпізнавання широко використовується для конвертації книг і документів в електронний вигляд, для автоматизації систем обліку в бізнесі або для публікації тексту на веб-сторінці. Оптичне розпізнавання тексту дозволяє редагувати текст, здійснювати пошук слова чи фрази, зберігати його в більш компактній формі, демонструвати або роздруковувати матеріал, не втрачаючи якості, аналізувати інформацію, а також застосовувати до тексту електронний переказ, форматування чи перетворення в мову. Оптичне розпізнавання тексту є досліджуваною проблемою в областях розпізнавання образів, штучного інтелекту та комп'ютерного зору.

Системи оптичного розпізнавання тексту вимагають калібрування для роботи з конкретним шрифтом; в ранніх версіях для програмування було необхідно зображення кожного символу, програма одночасно могла працювати тільки з одним шрифтом. В даний час найбільше поширені так звані "інтелектуальні" системи, з високим ступенем точності розпізнають більшість шрифтів. Деякі системи оптичного розпізнавання тексту здатні відновлювати початкове форматування тексту, включаючи зображення, колонки та інші нетекстові компоненти.

Інтерфейс онлайнового розпізнавання зазвичай складається з:

* ручки або стилуса, яким користувач здійснює введення
* поверхні, чутливої ​​до торкань, яка може бути інтегрована з дисплеєм
* програмного забезпечення, яке інтерпретує руху стилуса по друкарській поверхні, переводячи отримані лінії в цифровий текст.

Хоча розпізнавання рукописного тексту - засіб введення, до якого вже звикла громадськість, воно ще не досягло широкого поширення в настільних комп'ютерах і ноутбуках. Все ще вважається, що введення за допомогою клавіатури швидше і надійніше. Зараз існує безліч PDA, підтримуючих іноді і природний рукописний почерк, але точність все ще не є дуже високою, через що деякі люди знаходять навіть просту екранну клавіатуру більш ефективною.

Таким чином розробка системи, яка б допомогла вводити інформацію у компютер без допомоги клавіатури, і при цьому була зрозуміла будь-якій людині, без попередньої підготовки, є дуже актуальною проблемою.

**1.2 Існуючі рішення реалізації розпізнавання тексту**

У загальному випадку, розпізнавання тексту складається з наступних процедур і методів (рис. 1.1):

 попередня обробка;

 сегментація;

 розпізнавання.

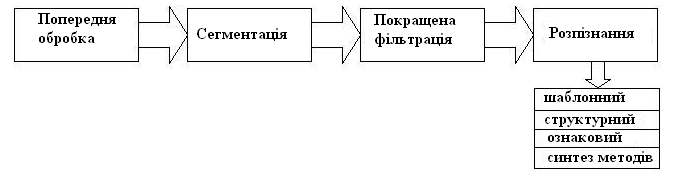


Рис. 1.1. Основні процедури й методи розпізнавання тексту

Процедура попередньої обробки використовується практично завжди після одержання інформації, і являє собою застосування операцій усереднення й вирівнювання гістограм, різного типу фільтрів для виключення перешкод, а також придушення зовнішніх шумів. Під сегментацією розуміється процес поділу зображення на окремі символи.

Кінцевий етап обробки - розпізнавання. Для цього етапу вхідними даними є зображення, отримані в результаті шумозаглушення й процесу сегментації.

Сьогодні відомо три підходи до розпізнавання символів: шаблоновий, структурний і ознаковий.

Шаблонові методи перетворять зображення окремого символу в растрове, порівнюють його з усіма шаблонами, наявними в базі й вибирають шаблон з найменшою кількістю крапок, відмінних від вхідного зображення. Шаблонові методи досить стійкі до дефектів зображення й мають високу швидкість обробки вхідних донних, але надійно розпізнають тільки ті шрифти, шаблони які їм "відомі". І якщо розпізнаваний шрифт хоч небагато відрізняється від еталонного, шаблонові методи можуть робити помилки навіть при обробці дуже якісних зображень.

У структурних методах об'єкт описується як граф, вузлами якого є елементи вхідного об'єкта, а дугами – просторові відносини між ними. Методи реалізуючий подібний підхід, звичайно працюють із векторними зображеннями. Структурними елементами є складовий символ лінії. Так, для букви "р" - це вертикальний відрізок і дуга.

До недоліків структурних методів варто віднести їхню високу чутливість до дефектів зображення, що порушує складові елементи. Також векторизація може додати додаткові дефекти. Крім того, для цих методів, на відміну від шаблонових і ознакових, дотепер не створені ефективні автоматизовані процедури навчання. Тому структурні описи найчастіше доводитися створювати вручну.

У ознакових методах усереднене зображення кожного символу представляється як об'єкт в n-мірному просторі ознак. Тут вибирається алфавіт ознак, значення яких обчислюються при розпізнаванні вхідного зображення. Отриманий n- мірний вектор рівняється з еталонними, і зображення ставиться до найбільш підходящому з них.

Також існує безліч методів, побудованих як синтез трьох підходів.

Нижче розглядаються самі популярні, добре вивчені й часто застосовні на практиці різні методи розпізнавання символів.

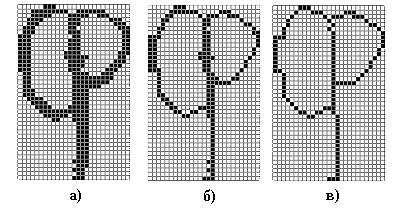
### 1.2.1 Розпізнавання скелетних образів

У першу чергу розпізнаваний символ піддається процедурі скелетизації (тоншання). Існують різноманітні методи одержання скелету символу, відмінні один від одного.

### 1.2.1.1 Метод Щепина.

Для кожного зовнішнього й внутрішнього контуру зображення перебувають вихідні верхні ліві крапки. Для чергової крапки контуру розглядається конфігурація восьми її сусідів. Крапка віддаляється, якщо вона не є кінцевою, і якщо після її видалення її сусіди як і раніше будуть утворювати зв'язну численість. Після аналізу крапки і її сусідів і можливого видалення крапки здійснюється перехід до наступної крапки контуру таким чином, щоб залишитися на границі зображення. Далі крок за кроком видаляється один шар крапок. Шари видаляється доти, поки не залишаться крапки, що не видаляють (рис.1.2).

Рис. 1.2. Скелетизація букви «Ф», що складає з одного зовнішнього й двох внутрішніх контурів, а) вихідне зображення; б) видалення одного шару; в) видалення другого шару

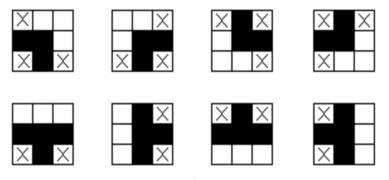


### 1.2.1.2 Скелетизація із застосуванням шаблонів

Для одержання кістякового зображення використаються шаблони, призначені для видалення зайвих пікселів, де знаком «X» відзначені пікселі будь-яких кольорів (рис. 1.3).

У будь-якій області, що відповідає одному із шаблонів, видаляється чорний центральний піксель. Здійснюється кілька проходів по зображенню, поки не залишиться пікселів, які мають бути видалені.

Рис.1.3. Шаблони скелетизації



Для підготовки до етапу розпізнавання в отриманому скелеті виділяємо ключові крапки: крапки з'єднання трьох або чотирьох ребер кістяка й кінцеві крапки.:

1. Створюємо порожній стек для зберігання координат початку й кінця ребер, крапок розгалуження скелету.

2. Заносимо в нього будь-яку крапку скелета.

3. Поки стек не порожній, продовжуємо кроки 4-7.

4. Вибираємо крапку зі стеку.

5. Будуємо послідовність ребер з обраної крапки зображення, поки не відбудеться розгалуження скелету, або не досягнемо кінцевої крапки.

6. Якщо досягли кінцевої крапки або досягли позначеного раніше ребра, то в масив заноситься пройдений шлях.

7. Якщо відбулося розгалуження скелету, то ми знайшли місце з'єднання ребер, і в масив заноситься послідовність ребер. У стек заносимо крапку розгалуження.

8. Переходимо до пункту 3.

В отриманому описі скелету виробляється огрубіла попередня обробка, що складається у видаленні коротких ліній й об'єднанні близьких тріодів.

### 1.2.2 Хвильовий метод

Метод полягає в аналізі шляху проходження сферичної хвилі по зображенню (рис. 1.4). На кожному кроці аналізується зсув центра мас крапок, що утворять нову генерацію хвилі, щодо його попередніх положень.

Метод складається з наступних кроків:

- побудова скелета зображення за допомогою сферичної хвилі;

- оптимізація отриманого кістяка.

Відстеження ліній зображення виробляється шляхом відстеження переміщення центра відрізка, утвореного крайніми крапками генерації хвилі (рис. 1.5, а). Після відстеження можливе згладжування відрізків.



Рис. 1.4. Проходження сферичної хвилі по зображенню

Виявлення збільшення «ширини» хвилі й поділи хвилі на дочірні дозволяє встановити крапку передбачуваного з'єднання двох відрізків. Визначення збільшення «ширини» хвилі виробляється шляхом порівняння «ширини» чергової генерації хвилі і її середнього значення за N попередніх генерацій (N задається заздалегідь). Причому ми одержуємо дві крапки (A,B) трасуючого відрізка. Після поділу хвилі на дві напівхвилі, ми одержуємо ще дві пари крапок (C,D) і (E,F). Крапка з'єднання відрізків лежить у шестикутнику ABCDEF і спочатку встановлюється як центр мас цього багатокутника (рис. 1.5,б). Корекція покладає на оптимізацію кістяка зображення.



Рис. 1.5. Відрізки, а) відстеження ліній зображення, б) визначення місця з'єднання відрізків

Отриманий кістяк зображення не є оптимальним. Це зв'язано, насамперед з тим, що ми маємо справу з растровим зображенням, а виходить, зображення має перекручування тим більші, ніж менше розмір зображення в символах.

Для зменшення впливу перекручувань на одержуваний кістяк необхідно його зробити оптимізацію. В одержуваному кістяку можливе подання одного відрізка деякою послідовністю ребер. Позбутися від цього можна аналізом послідовності ребер, оцінювання лінії, що виходить, від прямої. У випадку якщо відхилення перебуває в припустимих межах, необхідно послідовність ребер замінити на одне.

Для оптимізації кістяка проглядаються околиці виділених крапок з'єднання відрізків, тобто таких крапок, де спостерігається поділ хвилі на напівхвилі.



Рис. 1.6. Оптимізація крапки з'єднання відрізків

Для кожної особливої крапки отриманого кістякового подання обчислюється безліч топологічних ознак, основними з яких є:

• нормовані координати особливої крапки (вершина графа);

• довжина ребра до наступної вершини у відсотках від довжини всього графа;

• нормований напрямок з даної крапки на наступну особливу крапку;

• нормований напрямок входу в крапку, виходу із крапки;

• кривизна дуги, точніше "ліва" й "права" кривизна дуги, що з'єднує особливу крапку з наступною вершиною (кривизна ліворуч і праворуч). Кривизна обчислюється як відношення максимальної відстані від крапок дуги (що перебувають відповідно ліворуч/праворуч від прямої) до прямої, що з'єднує вершини, до довжини відрізка, що з'єднує ті ж вершини.,



Рис. 1.7. Приклад топологічних ознак

На рисунку 1.7 умовно показані деякі з топологічних ознак. Граф має п'ять особливих крапок - a0, a1, a2, a3, a4. При обході графа по маршруті a0 ==> a1 ==> a2... у вершині a1 умовно показані наступні ознаки: вектор R1 - напрямок входу в точку, вектор R2 - напрямок виходу із крапки, вектор R3 - глобальний напрямок на наступну особливу крапку. Двонаправлений вектор h показує величину "лівого" відхилення дуги (a1,a2) від прямої; "праве" відхилення дорівнює нулю.

Для деяких кодів число особливих крапок й, відповідно, число топологічних ознак занадто мало. Так, для коду, що відповідає символу "0", топологічних ознак взагалі ні, тому що немає ні однією особою крапки. Тому можуть обчислюватися й використатися наступні додаткові ознаки:

• розміри й положення компонентів і дір,

• "чорна" й "біла" ширина верхньої половини символу,

• модифіковані прямі прогини.

Прогини обчислюються як відстані від крапок скелетного подання до опуклої оболонки побудованого подання. Додатково запам'ятовується положення крапок максимального прогину. Для деяких топологічних кодів число топологічних ознак може бути досить велике, що може зажадати занадто великого набору еталонів для навчання, тому в ряді випадків у розпізнаванні використається частина ознак.

Символ визначається після порівняння його опису з кодами з бази даних, при цьому вибирається найближчий топологічний код.

Якщо символ після проходу циклу розпізнавання залишився нерозпізнаним, робиться спроба поліпшення зображення за допомогою наступних операцій:

• склеїти кінці ліній по напрямках (рис. 1.8); для цього розглянути напрямку всіх кінцевих дуг кістякового подання, і якщо напрямку яких-небудь ліній подібні (з точністю до знака), і вказують один на одного, можна спробувати їх з'єднати - можливо, це була суцільна лінія, що розірвалася внаслідок недостатнього рівня сканування або вихідних дефектів написання;

• склеїти крапки кістяка, що перебувають на мінімальній відстані одна від іншої;

• відкинути саму коротку лінію (дугу графа); зайві короткі дуги (лінії) нерідко виникають при рукописному написанні;



Рис. 1.8. Спроба поліпшення зображення символу. а) вихідне зображення символу; б) символ зі склеєними лініями

### 1.2.3 Фонтанне перетворення

Програмісти російської компанії ABBYY розробили оригінальні технології, що поліпшують якість розпізнавання. Ідея нового способу зберігання знань про букву – структурно-плямовий еталон, уперше з'явилася на світло в студентських роботах Д. Яна, К. Онисимовича й П. Сенаторова. Технологія розпізнавання за допомогою структурно-плямових еталонів одержала назву "фонтанне перетворення" (від англійського font - шрифт).

Фонтанне перетворення поєднує в собі переваги шаблонового й структурного методів і дозволяє уникнути недоліків, властивих кожному з них окремо. В основі цієї технології лежить використання структурно-плямовий еталона. Він дозволяє представити зображення у вигляді набору плям, зв'язаних між собою n-ими відносинами, що задають структуру символу. Ці відносини (тобто розташування плям одна відносно іншої) утворять структурні елементи, що становлять символ. Так, наприклад, відрізок - це один тип n-их відносин між плямами, еліпс – інший, дуга – третій. Інші відносини задають просторове розташування утворюючий символ елементів. Наочно це можна уявити собі у вигляді тенісних куль, нанизаних на гумовий джгут (рис. 1.9). Кулі можуть зрушуватися відносно один одного. Таке зв'язування рухливих куль можна "натягнути" на різні зображення одного символу, і система стає менш залежної від шрифтів і дефектів.

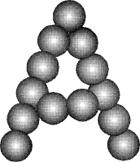


Рис. 1.9. Структурно-плямовий еталон букви «А»

В еталоні задаються:

 ім'я;

 обов'язкові, заборонні й необов'язкові структурні елементи;

 відносини між структурними елементами;

 відносини, що зв'язують структурні елементи із прямокутником, що описує, символу;

 атрибути, використовувані для виділення структурних елементів;

 атрибути, використовувані для перевірки відносин між елементами;

 атрибути, використовувані для оцінки якості елементів і відносин;

 позиція, з якої починається виділення елемента (відносини локалізації елементів).

Структурні елементи, виділювані для класу зображень, можуть бути вихідними і складовими. Вихідні структурні елементи – це плями, складові – відрізок, дуга, кільце, крапка. У якості складених структурних елементів, у принципі, можуть бути взяті будь-які об'єкти, описані в еталоні. Крім того, вони можуть бути описані як через вихідні, так і через інші складені структурні елементи.

Як відносини використаються зв'язки між структурними елементами, які визначаються або метричними характеристиками цих елементів , або їхнім взаємним розташуванням на зображенні.

При завданні структурних елементів і відносин використаються параметри, що конкретизують, що дозволяють довизначити структурний елемент або відношення при використанні цього елемента в еталоні конкретного класу. Для структурних елементів що конкретизують можуть бути, наприклад, параметри, що задають діапазон припустимої орієнтації відрізка, а для відносин – параметри, що задають граничне припустиму відстань між характерними крапками структурних елементів у відношенні стикаються.

Побудова й тестування структурно-плямових еталонів для класів розпізнаваних об'єктів, процес складний і трудомісткий. База зображень, що використається для налагодження описів, повинна містити приклади гарних і поганих (гранично припустимих) зображень для кожної графеми, а зображення бази розділяються на навчальну й контрольну численність.

Розроблювач опису попередньо задає набір структурних елементів (розбивка на плями) і відносини між ними. Система навчання по базі зображень автоматично обчислює параметри елементів і відносин. Отриманий еталон перевіряється й коректується по контрольній вибірці зображень даної графеми. По контрольній же вибірці перевіряється результат розпізнавання, тобто оцінюється якість підтвердження гіпотез.

Розпізнавання з використанням структурно-плямового еталона відбувається в такий спосіб. Еталон накладається на зображення, відносини між виділеними на зображенні плямами рівняються з відносинами плям в еталоні. Якщо виділені на зображенні плями й відносини між ними задовольняють еталону деякого символу, то даний символ додається в список гіпотез про результат розпізнавання вхідного зображення.

### 1.2.4 Адаптивне розпізнавання

Любий друкований текст має первинну властивість – шрифт, яким він надрукований. Із цього погляду існують два підходи до розпізнавання друкованого тексту: шрифтовой й безшрифтовий. Шрифтові або шрифтозалежні алгоритми використають апріорну інформацію про шрифт, яким надруковані букви. Це означає, що програмі розпізнавання повинна бути пред'явлена повноцінна вибірка тексту, надрукованого даним шрифтом. Програма вимірює й аналізує різні характеристики шрифту і заносить їх у свою базу еталонних характеристик. По закінченні цього процесу шрифтова програма оптичного розпізнавання символів готова до розпізнавання даного конкретного шрифту. Цей процес умовно можна назвати навчанням програми. Далі навчання повторюється для деякої безлічі шрифтів, що залежить від області застосування програми.

До недоліків даного підходу можна віднести наступні фактори:

 алгоритм повинен заздалегідь знати шрифт, що йому представляють для розпізнавання, тобто він повинен зберігати в базі різні характеристики цього шрифту. У реальності ж неможливо охопити всі шрифти і їхні модифікації. Інакше кажучи, цей фактор обмежує універсальність таких алгоритмів;

 для роботи програми розпізнавання необхідний блок настроювання на конкретний шрифт. Очевидно, що цей блок буде вносити свою частку помилок в інтегральну оцінку якості розпізнавання.

З іншого боку, у шрифтового підходу є перевага, завдяки якому його активно використають й, очевидно, будуть використати в майбутньому. А саме, маючи детальну апріорну інформацію про символи, можна побудувати досить точні й надійні алгоритми розпізнавання. Взагалі, при побудові шрифтового алгоритму розпізнавання надійність розпізнавання символу є інтуїтивно ясною й математично точно виразною величиною. Ця величина визначається як відстань у якому-небудь метричному просторі від еталонного символу, пред'явленого програмі в процесі навчання, до символу, що програма намагається розпізнати.

Другий клас алгоритмів – безшрифтові або шрифтонезалежні, тобто алгоритми, що не мають апріорних знань про символи, що надходять до них на вхід. Ці алгоритми вимірюють й аналізують різні характеристики (ознаки), властивим буквам як таким безвідносно шрифту й абсолютного розміру , яким вони надруковані. У граничному випадку для шрифтонезалежного алгоритму процес навчання може бути відсутнім. У цьому випадку характеристики символів вимірює, кодує й поміщає в базу програми сама людина. Однак на практиці, випадки, коли такий шлях вичерпно вирішує поставлене завдання, зустрічаються рідко. Більше загальний шлях створення бази характеристик полягає в навчанні програми на вибірці реальних символів.

До недоліків даного підходу можна віднести наступні фактори:

• реально досяжна якість розпізнавання нижче, ніж у шрифтових алгоритмів;

• варто вважати великою удачею, якщо безшрифтовий алгоритм володіє адекватним і фізично обґрунтованим, тобто природно виникає з основної процедури алгоритму, коефіцієнтом надійності розпізнавання.

Достоїнства цього підходу тісно пов'язані з його недоліками. Основними перевагами є наступні:

 універсальність. Це означає з однієї сторони застосовність цього підходу у випадках, коли потенційна розмаїтість символів, які можуть надійти на вхід системи, велика. З іншого боку, за рахунок закладеної в них здатності узагальнювати, такі алгоритми можуть екстраполювати накопичені знання за межі навчальної вибірки, тобто стійко розпізнавати символи, по виду далекі від тих, які були присутні в навчальній вибірці;

 технологічність. Процес навчання шрифтонезависимых алгоритмів звичайно є більше простим й інтегрованим у тому розумінні, що навчальна вибірка не розділена на різні класи (по шрифтах, кеглям і т.д.). При цьому відсутня необхідність підтримувати в базі характеристик різні умови спільного існування цих класів . Проявом технологічності є також той факт, що часто вдається створити майже повністю автоматизовані процедури навчання.

Розгляд обох підходів у порівнянні один з одним приводить до доцільності їхнього об'єднання (рис. 1.10). Ціль об'єднання очевидна - одержати метод, що сполучає одночасно універсальність і технологічність безшрифтового підходу й високу точність розпізнавання шрифтового.

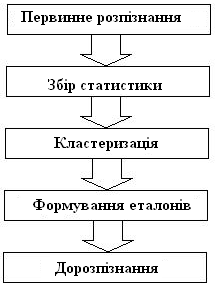


Рис. 1.10. Схема роботи адаптивного розпізнавання

При розробці об'єднаного методу розпізнавання, інформаційною одиницею, над якою повинен працювати метод, був обраний рівень однієї сторінки тексту. Так як це досить велика одиниця, для того щоб зібрана статистика була цілком представницька. Наприклад, кількість символів на звичайній машинописній сторінці тексту − 2000, відносна частота букви «н» в українській мові − 0,053; таким чином, на сторінці тексту кількість букв «н» у середньому становить 2000 х 0,053 = 106. Цього цілком достатньо для оцінки статистичних параметрів вибірки по даній букві, кластеризации й побудови двійкових еталонів для дорозпізнання.

Першим етапом є розпізнавання всієї сторінки якимсь готовим шрифтонезависимым алгоритмом із заданою якістю розпізнавання. Всі символи, розпізнані з надійністю, що перевищує заданий поріг, уважаються матеріалом для навчання бази характеристик.

Завданням кластеризации називається завдання розкласифікації пред'явлених об'єктів по декількох групах, причому число груп не обов'язково відомо. Кожну отриману групу часто називають кластером. Одним з методів рішення завдання кластеризации є метод ланцюгового розгорнення (рис. 1.11), коротко описаний нижче. У якості вихідного береться довільний об'єкт із пред'явленої сукупності, йому приписується номер 1 і відстань 0. Потім проглядаються всі об'єкти, що залишилися. Вибирається об'єкт, відстань від якого до вихідного мінімально. Йому привласнюється номер 2 і відповідна відстань. Потім серед тих, що залишилися, шукається об'єкт, відстань від якого до вже відзначеної безлічі об'єктів із двох елементів мінімально, і т.д. – завжди на наступному кроці вибирається об'єкт, відстань від якого до вже пронумерованих обєктів (як відстань до численості) мінімально, йому приписується черговий номер і ця відстань. Процедура повторюється доти, поки всі об'єкти не будуть пронумеровані.



Рис. 1.11. Приклад роботи ланцюгового розгорнення

Тепер для того, щоб розділити вихідну чисельність на кілька кластерів таким чином, щоб відстань між будь-якими об'єктами, що входять у різні кластери, була більше заданої відстані *d*, а для будь-яких об'єктів з одного кластера *(p1,p2)* можна було знайти об'єкти з того ж кластера (позначимо їх *o[1], o[2],…,o[n]*), такі що *o[1]=p1, o[n]=p2*, і для будь-якого *i<n* відстань між сусідніми об'єктами *d(o[i],o[i+1])* не більше *d*, досить просто переглянути всі приписані об'єктам відстані й позначити ті з них, які більше *d*. Нехай це будуть номери *N1, N2..,Nk*. Тоді до першого кластера варто віднести об'єкти з номерами менше *N1*, до другого всі об'єкти з номерами від *N1* до *N2* і т.д.

Розглянемо етап створення бази еталонних характеристик. Нагадаємо, що кластер складається з набору бітових растрів символів, які потрапили в цей кластер. На растрову сітку фіксованого розміру покладемо всі растри символів і просумуємо усередині кожного осередку. У відповідний осередок кластера запишемо суму (кількість разів, котре в цьому осередку зустрівся чорний піксель). Очевидно, що це число є частотою або, якщо його нормувати на кількість символів у кластері, імовірністю виникнення чорного пікселя в осередку. Осередку, у яких імовірність дорівнює нулю, заповнюються по-іншому. Туди записується відстань до найближчої крапки з ненульовий імовірністю. Ця відстань буде трактуватися як «негативна ймовірність» появи пікселю в цьому осередку.

Дорозпізнная є повторний прохід по всій сторінці з метою розпізнати все, що не розпізнано на першому проході, і одержати для всіх символів надійні оцінки точності розпізнавання. Схема процедури досить проста. На вхід надійшов черговий символ, що вимагає розпізнавання, він представлений у вигляді бітового растра. Порівнюючи цей символ з першим еталоном (кластером), одержуємо чисельну оцінку подібності символу з еталоном. Повторюємо порівняння з усіма іншими еталонами в базі. Після цього вибирається найкраща відповідь відповідно до отриманих оцінок.

### 1.2.5 Нейронні мережі

Нервова система й мозок людини складаються з нейронів, з'єднаних між собою нервовими волокнами. Нервові волокна здатні передавати електричні імпульси між нейронами. Розглянемо будову біологічного нейрона (рис. 1.12). Кожен нейрон має відростки нервових волокон двох типів –дендрити, по яких приймаються імпульси, і єдиний аксон, по якому нейрон може передавати імпульс. Аксон контактує с дендритами інших нейронів через спеціальні утворенняа - синапси, які впливають на силу імпульсу.

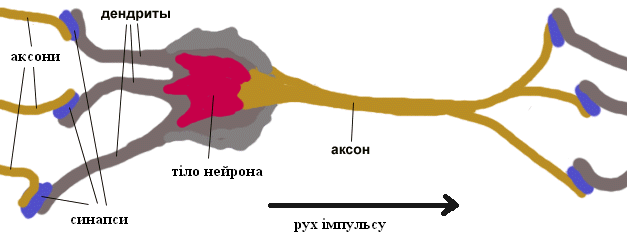


Рис. 1.12. Будова біологічного нейрона

Можна вважати, що при проходженні синапса сила імпульсу міняється в певне число разів, що ми будемо називати вагою синапса. Імпульси, що надійшли до нейрона одночасно по декількох дендритах, сумуються. Якщо сумарний імпульс перевищує деякий поріг, нейрон збуджується, формує власний імпульс і передає його далі по аксоні. Важливо відзначити, що ваги синапсів можуть змінюватися згодом, отже міняється й поводження відповідного нейрона.

На рисунку 1.13 побудована математична модель описаного процесу.

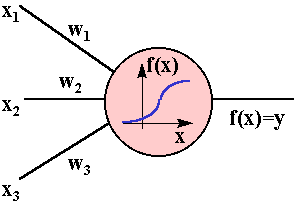


Рис. 1.13. Математична модель нейрона

Зображено модель нейрона із трьома входами (дендритами), причому синапси цих дендритів мають ваги w1, w2 , w3 . Нехай синапсам надходять імпульси з силами x1 , x2 , x3 відповідно, тоді після проходження синапсів і дендритів до нейрона надходять імпульси w1x1, w2x2, w3x3. Нейрон перетворить отриманий сумарний імпульс:

x=w1 x1 + w2 x2 + w3 x3, (1.2)

відповідно до деякої передатної функції f ( x) . Сила вихідного імпульсу дорівнює:

y= f ( x)= f(w1 x1+w2 x2+w3 x3 ) (1.3)

Таким чином, нейрон повністю описується своїми вагами wk і передатною функцією f ( x) . Одержавши набір чисел (вектор) xk як входи, нейрон видає деяке число y на виході.

Штучна нейронна мережа - це набір нейронів, з'єднаних між собою. Як правило, передатні функції всіх нейронів у нейронній мережі фіксовані, а ваги є параметрами нейронної мережі й можуть змінюватися. Деякі входи нейронів позначені як зовнішні входи нейронної мережі, а деякі виходи - як зовнішні виходи нейронної мережі. Подаючи будь-які числа на входи нейронної мережі, ми одержуємо якийсь набір чисел на виходах нейронної мережі. Таким чином, робота нейронної мережі складається в перетворенні вхідного вектора у вихідний вектор, причому це перетворення задається вагами нейронної мережі.

Практично будь-яке завдання можна звести до завдання, розв'язуваним нейронною мережею. У табл. 1.1 показано, яким чином варто сформулювати в термінах нейронної мережі завдання розпізнавання букв.

Таблиця 1.1. Формулювання завдання розпізнавання букв

|  |  |  |
| --- | --- | --- |
|  | Завдання розпізнавання  букв | Формулювання для нейронної мережі |
| Дано | Растрове чорно-біле  зображення літери розміром, приміром, 30х30 пікселів. | Вхідний вектор з 900 двійкових символів (900=30х30). |
| Треба | Визначити яка це літера  (в алфавіті 33 букви). | Побудувати нейронну мережу з 900 входами й 33 виходами, які позначені буквами. Якщо на вході зображення букви «А», то максимальне значення вихідного сигналу досягається на виході «А». Аналогічно мережа працює для всіх 33 букв. |

### 1.2.5.1 Побудова нейронної мережі

Тепер, коли стало ясно, що саме ми хочемо побудувати, ми можемо переходити до питання "як будувати таку нейронну мережу". Це питання вирішується у два етапи:

1. Вибір типу (архітектури) нейронної мережі.

2. Підбір ваг (навчання) нейронної мережі.

На першому етапі потрібно вибрати наступне:

 які нейрони ми хочемо використати (число входів, передатні функції);

 яким чином варто з'єднати їх між собою;

 що взяти як входи й виходи нейронної мережі.

Існує кілька десятків різних нейромережевих архітектур, причому ефективність багатьох з них доведена математично. Найбільш популярної й вивчені архітектури - це багатошаровий перцептрон, нейронна мережа із загальною регресією, нейронні мережі Кохонена та інших.

На другому етапі нам треба "навчити" обрану нейронну мережу, тобто підібрати такі значення її ваг, щоб вона працювала належним чином. Ненавчена нейронна мережа подібна до дитини – її можна навчити чому завгодно. У використовувані на практиці нейронних мережах кількість ваг може становити кілька десятків тисяч, тому навчання – дійсно складний процес. Для багатьох структур розроблені спеціальні алгоритми навчання, які дозволяють настроїти ваги нейронної мережі певним чином. Найбільш популярний із цих алгоритмів - метод зворотного поширення помилки (Error Back Propagation), використовуваний, наприклад, для навчання перцептрона.

### 1.2.5.2 Навчання нейронної мережі

Навчити нейронну мережу – виходить, повідомити їй, чого ми від неї домагаємося. Цей процес дуже схожий на навчання дитини алфавіту. Показавши дитині зображення букви «А», ми запитуємо його: «Яка це буква?» Якщо відповідь невірна, ми повідомляємо дитині та відповідь, що ми хотіли б від нього одержати: «Це буква А». Дитина запам'ятовує цей приклад разом з вірною відповіддю, тобто в його пам'яті відбуваються деякі зміни в потрібному напрямку. Ми будемо повторювати процес пред'явлення букв знову й знову доти, доки всі 33 букви будуть твердо запам’ятовано. Такий процес називають «навчання з вчителем».

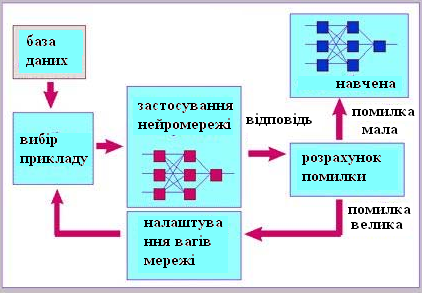


Рис. 1.14. Процес навчання нейронної мережі

При навчанні нейронної мережі ми діємо зовсім аналогічно. У нас є деяка база даних, що містить приклади (набір рукописних зображень букв). Пред'являючи зображення букви «А» на вхід нейронної мережі, ми одержуємо від неї деяку відповідь, не обов'язково вірну. Нам відомий і вірна відповідь – у цьому випадку нам хотілося б, щоб на виході нейронної мережі з міткою «А» рівень сигналу був максимальний. Звичайно як бажаний вихід у завданні класифікації беруть на бір(1, 0, 0, ...), де 1 стоїть на виході з міткою «А», а 0 - на всіх інших виходах. Обчислюючи різницю між бажаною відповіддю й реальною відповіддю мережі, ми одержуємо 33 числа - вектор помилки. Алгоритм зворотного поширення помилки - це набір формул, що дозволяє по векторі помилки обчислити необхідні виправлення для ваг нейронної мережі. Ту саму букву (а також різні зображення однієї й тієї ж букви) ми можемо пред'являти нейронної мережі багато разів. У цьому змісті навчання скоріше нагадує повторення вправ у спорті - тренування.

Виявляється, що після багаторазового пред'явлення прикладів ваги нейронної мережі стабілізуються, причому нейронна мережа дає правильні відповіді на всі (або майже всі) приклади з бази даних. У такому випадку говорять, що «нейронна мережа вивчила всі приклади», «нейронна мережа навчена», або «нейронна мережа натренована». У програмних реалізаціях можна бачити, що в процесі навчання величина помилки (сума квадратів помилок по всіх виходах) поступово зменшується. Коли величина помилки досягає нуля або прийнятного малого рівня, тренування зупиняють, а отриману нейронну мережу вважають натренованої і готової до застосування на нових даних.

Важливо відзначити, що вся інформація, що нейронна мережа має про завдання, утримується в наборі прикладів. Тому якість навчання нейронної мережі прямо залежить від кількості прикладів у навчальній вибірці, а також від того, наскільки повно ці приклади описують дане завдання. Так, наприклад, безглуздо використати нейронну мережу для пророкування фінансової кризи, якщо в навчальній вибірці криз не представлено. Вважається, що для повноцінного тренування нейронної мережі потрібно хоча би кілька десятків (а краще сотень) прикладів.

### 1.2.5.3 Застосування нейронної мережі

Після того, як нейронна мережа навчена (рис. 1.15), ми можемо застосовувати її для рішення корисних завдань. Найважливіша особливість людського мозку полягає в тому, що, один раз навчившись певному процесу, він може вірно діяти й у тих ситуаціях, у яких він не бував у процесі навчання. Наприклад, ми можемо читати майже будь-який почерк, навіть якщо бачимо його перший раз у житті. Так само й нейронна мережа, грамотним образом навчена, може з великою ймовірністю правильно реагувати на нові, не пред'явлені їй раніше дані. Наприклад, ми можемо намалювати букву «А» іншим почерком, а потім запропонувати нашій нейронній мережі класифікувати нове зображення. Ваги навченої нейронної мережі зберігають досить багато інформації про подібність і розходження букв, тому можна розраховувати на правильну відповідь і для нового варіанта зображення.

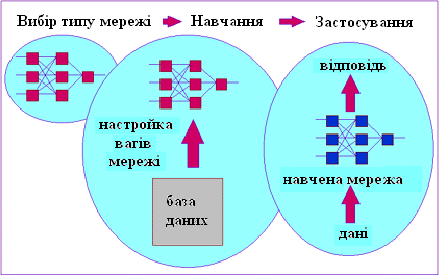


Рис. 1.15. Застосування нейронної мережі

**1.3 Огляд існуючих програмних рішень**

Серійні пристрої, в яких в якості альтернативи традиційному введенню за допомогою клавіатури використовувався введення рукописного тексту з'явилися на початку 1980-х. Це були, наприклад, термінали з рукописним введенням, такі як Pencept Penpad і Inforite point-of-sale terminal. З розширенням ринку персональних комп'ютерів з'явилися кілька комерційних продуктів, покликаних замінити клавіатуру і мишу на персональних комп'ютерах єдиною системою, що надаються PenCept, CIC та інші. Першим, що знаходяться у відкритому продажі, планшетним комп'ютером був GRiDPad від GRiD Systems, випущений у вересні 1989. Його операційна система була заснована на MS-DOS.

На початку 90-х виробники апаратного забезпечення, включаючи NCR, IBM і EO, випустили планшетні комп'ютери з операційною системою PenPoint, розробленої GO Corp. PenPoint використовувала рукописне введення всюди і забезпечувала сумісність зі стороннім програмним забезпеченням. Планшетний комп'ютер IBM був першим використовують ThinkPad і розпізнавання рукописного тексту IBM. Ця система розпізнавання була пізніше портована в Microsoft Windows for Pen Computing і IBM's Pen для OS / 2. Жодна з цих технологій не мала комерційного успіху.

Розвиток електроніки дозволили обчислювальної потужності, необхідної для розпізнавання рукописного введення, уміщатися в менші розміри, ніж у планшетних комп'ютерів, і використовувати розпізнавання рукописного тексту в якості засобу введення для PDA. Першим PDA, забезпечує письмовий введення, був Apple Newton, який продемонстрував громадськості гідності такого роду користувальницького інтерфейсу. Однак пристрій не було комерційно вдалим внаслідок недосконалості програмного забезпечення, яке повинно було намагатися вивчити манеру письма користувача. Після розриву з Apple Newton, технологія була портована в Mac OS X 10.2 і більш пізні у вигляді технології Inkwell (Macintosh).

Сучасна система розпізнавання рукописного тексту включається в операційні системи Microsoft, використовувані на планшетних комп'ютерах. Вона заснована на TDNN-класифікаторі, названому "Inferno", створеному в Microsoft. Пізніше версія CalliGrapher, що розпізнає рукописних введення програмного забезпечення, використовуваного в Newton OS 2.0, була включена в якості вторинного розпізнавача. Нове покоління CalliGrapher в даний час розробляється для Windows Mobile корпорацією PhatWare.

Технологія розпізнавання рукописного введення "третього покоління" riteScript, розроблена корпорацією EverNote в 2000-2004, включається в ritePen and EverNote. ritePen також включає технологію, що дозволяє комбінувати riteScript з вбудованим розпізнаванням рукописного тексту Windows Vista, щоб поліпшити точність розпізнавання кожного движка розпізнавання рукописного тексту.

Планшетний ПК - це ноутбук, забезпечений дігітайзером і стилусом, що дозволяє користувачеві писати від руки текст на екрані. Операційна система розпізнає рукописний текст і конвертує його в машинний текст. Windows Vista і Windows 7 включають можливості персоналізації, які вивчають особливості почерку користувача і словники для англійської, японської, китайської традиційного, китайського спрощеного, корейського та ін мов. (Windows 7 може розпізнавати рукописний текст російською мовою.) "Майстер персоналізації" дозволяє перенавчити систему для більш високої точності розпізнавання рукописного тексту, написаного певним користувачем комп'ютера. Ця система відрізняється від менш розвинутої системи розпізнавання рукописного тексту, використовуваної в Windows Mobile.

Сучасні програми з OCR вміють:

* розпізнавати тексти, набрані не лише різними шрифтами, але написані екзотичними, у тому числі і рукописними
* коректно працювати з текстами, які містять слова на декількох мовах
* розпізнавати таблиці
* розпізнавати нечітко набрані чи написані тексти

Еволюція OCR-програм дуже стрімка. Сьогодні це одна з галузей ринку, що швидко розвиваються, програмних продуктів. Перші шрифтонезалежні системи (1994 р. для російської мови) дозволяли якісно читати тільки одномовний текст і зберігати результат розпізнавання в TXT-файлі. У системах випуску 1995 р. документ зберігається вже у форматі RTF, у вигляді, схожому на оригінал (зберігання інформації про шрифти, абзаци, ілюстрації). Програми цього року дозволяли сполучити в інформації англійську і російську мови. Системи 1996 р. розпізнають таблиці, бланки, анкети, можуть працювати в мережному режимі і допускають довільну комбінацію базових мов в одному тексті (FineReader 3.0).

Але головною причиною такого сканерного буму стала гучна конкуренція двох систем оптичного розпізнавання символів - однієї, за назвою CuneiForm, виробництва компанії Cognitive Technologies, і іншої - FineReader від Bit Software.

Історія самого масового попиту на системи OCR у Росії почалася з «протиборства» систем CuneiForm і FineReader однакових версій 1.3. На думку багатьох незалежних спеціалістів, CuneiForm по сумі показників був тоді сильніше. Кістяк колективу розроблювачів цієї програми базувався в США. Але, на жаль, ще до виходу версії CuneiForm 2.0 ця команда практично перестала існувати. А BIT свій колектив програмістів зберегла. Можливо, саме цей факт послужив головною причиною того, що результати тестування незалежними російськими виданнями наступних версій цих програм були однозначні: FineReader 2.0 істотно випереджав CuneiForm 2.0.

Після версій 2.0 Cognitive Technologies випустила версії CuneiForm 2.95 (під Windows 95) і 2.96, у якій був перероблений алгоритм розпізнавання. BIT Software пізніше представила версію FineReader 3.0.

Основні можливості FineReader 3.0:

* Розпізнавання текстів на російській, українській (при використанні модуля Ukraine Add-On), англійській, німецькій, французькій мовах, а також змішаних текстів, у яких допускається довільне сполучення цих мов;
* Наявність умонтованої програми перевірки орфографії Lingvo Corrector, що можна підключити до Microsoft Word for Windows 95;
* Збереження структури документа у форматі RTF;
* Розпізнавання таблиць (можна завантажити готову таблицю в Microsoft Word або Exсel);
* Розпізнавання складних форм із збереженням у форматі бази даних (тільки у версії Fine Reader 3.0 Professional);
* Можливість навчання новим мовам (тільки у версії FineReader 3.0 Professional).

Сьогодні OCR-системи дозволяють уводити друкарські тексти різної якості - надруковані на лазерному, струменевому або матричному принтерах, книги, листівки, газети, факс.

Навчати шрифтам систему немає необхідності, тому що усі популярні системи розпізнавання являються шрифтонезалежними (омнішрифтовими). Більш того, у результуючому документі будуть відбитий тип шрифту (із засічками, типу Times; типу Halvetica; моноширинний, типу Courier), а також висота і стиль написання символів (жирні, похилі, підкреслені).OCR-програма, звісно ж, не зуміє розпізнати документ, що складно прочитати, але більшість документів, що нормально читаються людиною, можна ввести автоматично.