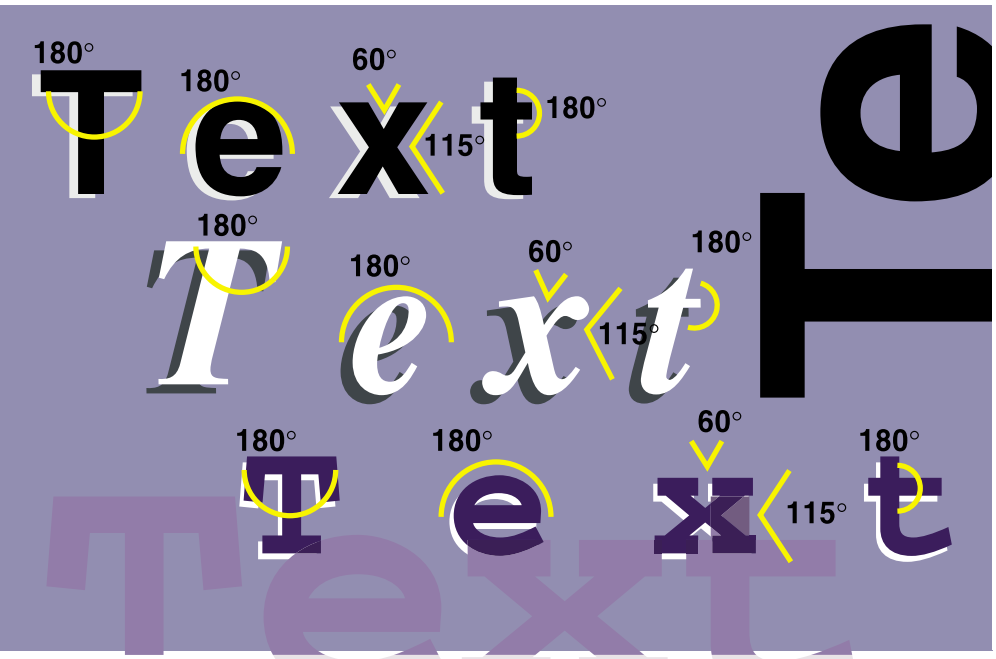


Charakterystyczne współczynniki kątowe, kształty krzywizn oraz inne parametry pomiarowe i reguły pomagają programom we właściwym rozpoznawaniu liter

Pecet uczy się czytać

Programy do rozpoznawania tekstu nie tylko, w sposób podobny do człowieka, analizują odczytywane znaki za pomocą sztucznych sieci neuronowych, ale również samodzielnie prowadzą dyskusje naukowe. W trudniejszych przypadkach umieszczone w tych aplikacjach systemy ekspertowe dyskutują bowiem uzyskane przez siebie „wyniki badań". Kontrolę nad przebiegiem takiej konferencji sprawuje jej najbardziej inteligentny uczestnik – logika rozmyta.

Mimo dynamicznego rozwoju publikacji elektronicznych i Internetu papier, wynaleziony przez Chińczyków prawie dwa tysiące lat temu, nadal jest najchętniej używanym medium do przekazywania „słowa pisanego". Dlatego też od początku rozwoju techniki komputerowej pojawił się problem przenoszenia informacji z komputera na papier – drukowania, a także odczytywania informacji z papieru i przekładania jej na język zrozumiały dla przeciętnego peceta. Rozwiązywaniem drugiego z wymienionych zadań, czyli rozpoznawaniem tekstu, zajmują się systemy typu OCR (*Optical Character Recognition* – optyczne rozpoznawanie znaków). W tej chwili systematycznie zastępują je młodsi bracia, tak zwane systemy ICR (ang. Intelligent Character Recognition – inteligentne rozpoznawanie znaków), których „inteligencja" polega na wykorzystywaniu algorytmów rozpoznawania wzorowanych na ludzkim sposobie percepcji.

Z całą pewnością żadnego użytkownika komputera nie trzeba dzisiaj zachęcać do używania drukarki, natomiast wykorzystywanie skanera i oprogramowania OCR nie jest jeszcze zjawiskiem powszechnym. Wyobraźmy sobie zatem taką sytuację: jest późne, piątkowe popołudnie, właśnie przygotowujemy się do wyjścia z biura, gdy nagle otrzymujemy od szefa polecenie natychmiastowego opracowania dziesięciostronicowej umowy (czy też cennika, oferty, artykułu, informacji prasowej itp.) dla jednego z kontrahentów. Jak to w życiu bywa, wzorcowy tekst umowy „wyparował" nam z komputera podczas ostatnich porządków na dysku twardym i pozostała jedynie jego papierowa wersja. Cóż więc możemy z tym fantem zrobić? Pierwsze rozwiązanie to wyjaśnienie szefowi problemów z wywiązaniem się z polecenia i sugestia, iż dopiero w przyszłym tygodniu sprawa zostanie załatwiona – jest to rozwiązanie nieeleganckie i na pewno nie wpłynie pozytywnie na naszą karierę zawodową. Drugi pomysł to ręczne „wklepanie" od początku całego tekstu. W tym wypadku sukces jest gwarantowany, umowa zostanie przygotowana, ale prawdopodobnie wyjdziemy z biura późnym wieczorem. Ponadto, przepisując tekst z kartki możemy popełnić wiele pomyłek, których korekta może potrwać długo. Najlepszym wyjściem jest rozwiązanie trzecie – jeżeli w biurze mamy skaner i program OCR, możemy szybko zeskanować wydruk umowy (skanowanie jednej strony A4 trwa, w zależności od skanera, ok. 20–40 sekund), a następnie automatycznie przekonwertować obraz strony na tekst i zapisać w formacie używanym przez nasz edytor. Proces konwersji, czyli rozpoznawania tekstu, w zależności od jakości wydruku, wielkości czcionki, typu komputera oraz oczywiście rodzaju oprogramowania OCR może zająć od 30 sekund do kilku minut na jedną stronę. Przy

odrobinie wprawy całkowity czas wczytania umowy wraz z jej korektą i przygotowaniem poprawek dla szefa nie powinien, w naszym przypadku, przekroczyć godziny! Dziesięć stron tekstu to około 4500 wyrazów; w najgorszym przypadku daje to prędkość rozpoznawania i korekty równą 75 słowom na minutę, czego pozazdrościć nam może nawet bardzo wprawna maszynistka. W nagrodę mamy wolne całe piątkowe popołudnie i uznanie szefa...

Krok po kroku

Nowoczesne oprogramowanie ICR nie realizuje wyłącznie rozpoznawania pojedynczych znaków. Proces konwersji zdjęcia na tekst przebiega w czterech etapach, z których każdy wykorzystuje zaawansowane algorytmy, niejednokrotnie opracowane na bazie skomplikowanych teorii naukowych. Skuteczność działania każdego z etapów wpływa znacząco na jakość danego programu ICR.

Pierwszy etap to wstępne przetworzenie obrazu strony (*Preprocessing*). Na tym etapie automatycznie wykrywana i korygowana jest orientacja tekstu na stronie – program sprawdza, czy wydruk nie został włożony do skanera „do góry nogami" lub koryguje często występujące „przekręcenie tekstu" (gdy wiersze z tekstem nie są równoległe do podstawy dokumentu). Również w tym kroku program ustala, czy ma do czynienia z drukiem „czarno na białym" czy też wydruk jest negatywem – białe litery na czarnym tle. Podczas przetwarzania wstępnego niektóre programy ICR wykonują różnego typu filtracje, usuwając np. drobne kropki i zakłócenia, co może wpłynąć na poprawę skuteczności następnych etapów.

W trakcie segmentacji (*Auto Zoning* lub *Page Decomposition*), drugiego etapu przetwarzania, program automatycznie wykrywa te fragmenty obrazu, które warto rozpoznawać. Rezultatem tej operacji jest wyróżnienie w obrazie dokumentu obszarów zawierających tekst (*Text Zones*), grafikę lub zdjęcia (*Graphic Zones*) oraz tabelek (*Table Zones*). Dodatkowo ustalana jest kolejność obszarów tekstowych, tak aby wynik rozpoznawania jak najdokładniej odzwierciedlał logiczny porządek tekstu w skanowanym dokumencie. Regułą jest to, że użytkownik może manualnie skorygować rezultat segmentacji.

Kolejny etap to rozpoznawanie znaków (*Character Recognition*). Podczas rozpoznawania obraz dokumentu, wiersz po wierszu, znak po znaku, zamieniany jest na tekst. Nowoczesne programy ICR wykorzystują co najmniej dwie, równolegle działające metody konwersji obrazu na znaki, a ostateczny rezultat tejże konwersji weryfikuje i ustala specjalny system ekspertowy. W rozpoznawaniu może aktywnie uczestniczyć użytkownik, „podpowiadając" komputerowi wówczas, gdy ten ma kłopot z rozpoznaniem określonego znaku. „Podpowiedź" taką program zapamiętuje i wykorzystuje przy przetwarzaniu kolejnych fragmentów tekstu, gdy napotka podobnie wyglądający znak.

Ostatnim krokiem jest analiza językowa (*Language Analysis*). W prostszych systemach OCR analiza językowa jest całkowicie oddzielona od etapu rozpoznawania znaków i sprowadza się do wykrywania literówek (Spell Checking) na podstawie słownictwa wybranego języka. W programach ICR algorytmy analizy językowej uczestniczą w rozpoznawaniu znaków i pełnią funkcję ekspertów przejmujących odpowiedzialność za ostateczny wynik, a wykorzystują do tego wiedzę zarówno o słownictwie, jak i o gramatyce.

Od wzorca do sieci neuronowej

Podstawowe algorytmy rozpoznawania znaków drukowanych funkcjonują już od ponad 25 lat. Początkowo był to stosunkowo prosty, sprzętowy mechanizm polegający na porównywaniu pojedynczych liter z ich odpowiednikami na liście wzorców. W połowie lat sześćdziesiątych wprowadzone zostały w tym celu dwa znormalizowane zestawy znaków: OCR-A i OCR-B. Jeszcze dzisiaj tego typu czcionki są stosowane w polach kodowych czeków bankowych.

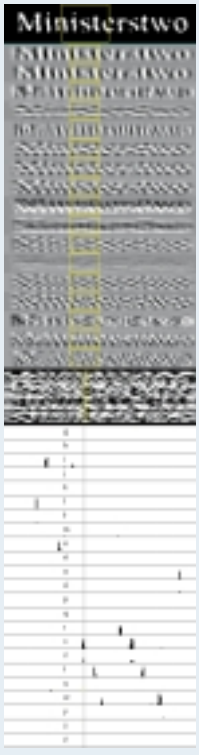
Jeśli odczytany obraz znaku będzie zgodny z obiektem wzorcowym, przechowywanym w pamięci w postaci macierzy pikseli, to jest mu przyporządkowywany odpowiedni kod ASCII i daną literę uznaje się za rozpoznaną. W praktyce oznacza to, że każda litera A, B lub C jest identyfikowana tylko wówczas, gdy wygląda identycznie jak odpowiadający jej znak wzorcowy. W przeciwnym wypadku litera taka będzie ignorowana bądź sygnalizowana jako błąd.

Później na rynku pojawiły się bardziej zaawansowane urządzenia odczytujące, które oprócz znaków OCR-A i OCR-B potrafiły także rozpoznać standardowe pisma maszynowe. Nadal jednak do tego celu była wykorzystywana technika „porównywania ze wzorcem" (*Matrix Matching*), polegająca na porównywaniu rozpoznawanego znaku z zapamiętanym, matrycowym wzorcem czcionki. Wystarczyły niewielkie odstępstwa kształtu czcionki od zdefiniowanego wzorca, aby skutecznie uniemożliwić rozpoznanie tekstu.

▶230

podstawy

Przykład „bezsegmentacyjnego" rozpoznawania słowa „Ministerstwo" z użyciem sieci neuronowej typu *Spatiognitron*.
a) obraz podawany na wejście sieci
b) mapy prostych cech znaków, wykrytych przez sieć (17 różnych map)
c) mapy cech złożonych (36 map)
d) analogowe wyjście sieci – neurony odpowiadające znakom od „g" do „ż". Im wyższy słupek na zadanej pozycji w wierszu tym bardziej prawdopodobne rozpoznanie określonego znaku.
Żółte prostokąty wyznaczają zbiór danych, na podstawie których sieć neuronowa rozpoznała obecność w obrazie wejściowym litery „s".



Dopiero od 1975 roku technika porównywania ze wzorcem została zastąpiona techniką „analizy cech charakterystycznych" (Feature Recognition). W przypadku analizy szczegółów typograficznych każdej litery nie dokonuje się porównań z zapamiętanymi obrazami czcionek, lecz poszukuje się charakterystycznych kształtów krzywizn, współczynników kątowych, elementów kolistych oraz proporcji między liniami poprzecznymi i podłużnymi. Z uwagi na fakt, że matematyczno-geometryczne opisy liter są w zdecydowanej większości przypadków niezmienne (np. litera „p" składa się zazwyczaj z długiej, pionowej kreski „|" i kółeczka „o" przyklejonego do tejże kreseczki po jej prawej stronie, w górnej części), taka metoda może być stosowana do rozpoznawania różnego typu czcionek i różnych wielkości pisma. Dlatego też technika ta nosi nazwę metody „omnifont" (łac. omni – wszyscy). Pierwszym urządzeniem, które korzystało z opisanej wyżej metody, był wdrożony w 1975 roku czytnik KRM (*Kurzweil Reading Machine*) nazwany tak na cześć swojego twórcy – Raymonda Kurzweila.

Początek lat osiemdziesiątych zaowocował nagłym rozwojem nowej dziedziny nauki jaką była teoria sztucznych sieci neuronowych (*Artificial Neural Network Theory* lub *Neurocomputing*). Niemal natychmiast sieci neuronowe znalazły zastosowanie w algorytmach rozpoznawania tekstu.

Początkowo ANN (*Artificial Neural Networks*) wykorzystywane były wyłącznie jako klasyfikatory; na podstawie dostarczonej na wejście informacji o matematyczno-geometrycznych cechach znaku podejmowały decyzję o tym, jaki to znak. Zaletą sieci neuronowej, w porównaniu z klasyczną techniką analizy cech charakterystycznych, była możliwość uczenia się „na przykładach". Wystarczyło „przeszkolić" sieć, prezentując jej treningowy zestaw czcionek, a ta sama wyrabiała sobie „ogólną" wiedzę na temat kształtów poszczególnych znaków.

We współczesnych systemach ICR sieci neuronowe stosowane są wówczas, gdy zawodzą inne techniki rozpoznawania znaków. Architektura sieci neuronowej do rozpoznawania pisma wzorowana jest na strukturze połączeń występujących w tej części mózgu człowieka, która zajmuje się przetwarzaniem informacji wizualnej (*Visual Cortex*). Sieć nie tylko klasyfikuje cechy, ale także sama je wykrywa – specjalnie połączone warstwy neuronów są w stanie, w trakcie uczenia, „wyłuskać" z pikselowego obrazu zadanego znaku cechy geometryczne odróżniające go od innych liter. Dodatkowo niektóre modele ANN potrafią skutecznie rozpoznawać całe sekwencje mocno zdeformowanych i zakłóconych znaków bez konieczności podziału sekwencji na pojedyncze litery. Jest to tak zwane rozpoznawanie „bezsegmentacyjne" (ang. Segmentation Free Method), w którym stosuje się techniki przetwarzania podobne do stosowanych w rozpoznawaniu mowy.

Mimo wielu zalet zaawansowane ANN mają wady wykluczające, przynajmniej na razie, możliwość zastosowania sieci neuronowej jako jedynego algorytmu rozpoznawania. Jedną z podstawowych wad jest relatywnie długi czas wyliczania stanu sieci. Przykładowo, skonstruowany na początku lat dziewięćdziesiątych i wzorowany na budowie mózgu Neocognitron – sieć do rozpoznawania pojedynczych znaków alfanumerycznych – złożony był z 70 045 neuronów (elementarnych jednostek obliczeniowych) i czas wyliczania odpowiedzi sieci, po wprowadzeniu na wejście jednego znaku, przekraczał trzy minuty! Do symulacji Neocognitronu używano wówczas komputera typu Sun SparcStation.

Praktyczne implementacje rozbudowanych sieci neuronowych są obecnie bardzo mocno optymalizowane i często wykorzystują najnowsze osiągnięcia techniki mikroprocesorowej (np. w przypadku symulatorów specyficzne właściwości arytmetyki zmiennoprzecinkowej). Jednak prędkość rozpoznawania „neuronowego" jest średnio o rząd mniejsza niż prędkość rozpoznawania metod klasycznych typu „omnifont".

W komercyjnych programach ICR w rozpoznawaniu znaków uczestniczy kilka algorytmów, a w zależności od jakości wydruku udział każdego z nich może być różny. Dla tekstu pisanego ładnym, czytelnym drukiem stosuje się szybkie rozpoznawanie „omnifont". W tekście gorszej jakości niektóre wiersze, lub ich części, mogą być rozpoznawane przez bardziej rozbudowane modyfikacje algorytmu „omnifont" bądź sieci neuronowe. W każdym z wymienionych przypadków stosowane jest także proste porównywanie ze wzorcem, dzięki któremu użytkownik może „uczyć" program, wprowadzając do rozpoznawania np. nietypowe znaki.

W niepewności siła?!

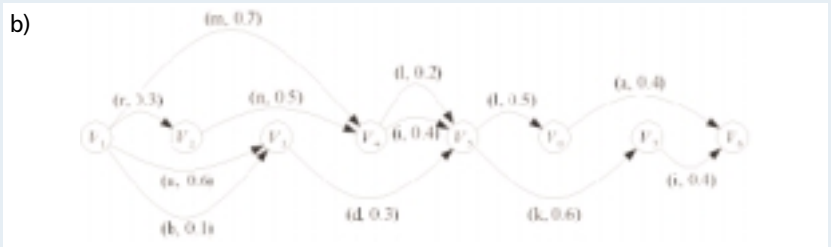
Komputer to urządzenie, które w trakcie obliczeń posługuje się bardzo prostym i klarownym systemem liczbowym – klasyczną logiką dwuwartościową. Coś może być „prawdą" lub „fałszem" i wtedy odpowiedni bit we właściwym rejestrze lub komórce pamięci przyjmuje wartość „0" lub „1". Każdy, najbardziej elementarny proces obliczeniowy polega na przekształcaniu szeregu wartości typu „prawda" ▶233

podstawy

a)

Przykład wariantowego rozpoznawania słowa „mila":
a) obraz wejściowy, b) rezultat działania algorytmów rozpoznawania znaków – graf wszystkich możliwych wariantów.
Spośród wszystkich możliwych słów, które reprezentuje graf b) w języku polskim poprawne są słowa „mila" i „miki". Biorąc pod uwagę prawdopodobieństwa rozpoznawania poszczególnych znaków ostatecznym wynikiem rozpoznawania będzie słowo „miki".

b)



podstawy

a)

b)

c)

Przykłady wierszy z tekstem, dla których trzeba zastosować bardziej skomplikowane algorytmy rozpoznawania znaków, np. sieci neuronowe. a) „porwane litery", b) „sklejenia" c) „zakłócenia".

i „fałsz" na inny tego typu szereg, według ściśle określonych i zarazem prostych reguł logicznych. Nawet skomplikowane operacje zmiennoprzecinkowe są w istocie żonglowaniem wartościami typu „prawda" i „fałsz".

W odróżnieniu od tego typu działań w naturalnych i sztucznych sieciach neuronowych proces przetwarzania informacji podlega tzw. logice rozmytej (*fuzzy logic*, fuzzy = niewyraźny, zamazany), w której funkcjonuje nie tylko pojęcie „prawdy" i „fałszu", ale także występują pojęcia pośrednie typu „częściowa prawda" lub „nie do końca fałsz". W istocie przy takich założeniach wynik dedukcji logicznej przyjmuje wartość analogową z zakresu od „0" do „1", przy czym „0" oznacza „absolutny fałsz", a „1" wskazuje na „całkowitą prawdę".

Logika rozmyta wykorzystuje fakt, iż ludzki mózg potrafi sensownie rozwiązywać problemy nawet wtedy, gdy nie ma jednoznacznie określonych parametrów. W procesie podejmowania decyzji korzysta on bowiem z takich wartości przybliżonych, jak „mniej więcej", „prawie" czy też „w znacznym stopniu".

Podobnie jak człowiek „rozumują" również systemy ekspertowe bazujące na logice rozmytej, umieszczone w programach ICR. Działają one w sposób analogowy, a nie cyfrowy – znają więc nie tylko słowa „tak" i „nie", ale także „dużo" i „mało". Bezpośrednim rezultatem wykorzystania logiki rozmytej jest możliwość tworzenia wielu wariantów rozpoznawania, np. jeżeli znak został rozpoznany na 60% jako litera „F", to być może na 30% jest to litera „E", ewentualnie litera „P" na 10%. Co więcej, każdy algorytm uczestniczący w rozpoznawaniu znaku może postawić różne hipotezy co do prawdopodobieństwa wystąpienia każdego z wariantów, a wiarygodność tychże hipotez musi ocenić dodatkowy ekspert. W efekcie wynik rozpoznawania jednego wiersza tekstu to graf opisujący wszystkie możliwe sekwencje znaków z prawdopodobieństwem wystąpienia znaku na zadanej pozycji.

Konferencja lingwistów

Tam, gdzie pojawia się wiele wariantów, mamy do czynienia z problemem wyboru tego „najlepszego". Dzięki zastosowaniu logiki rozmytej i sztucznych sieci neuronowych dla każdego wiersza z tekstem możemy uzyskać wiele różnych sekwencji słów. Powszechnie stosowaną i dającą bardzo dobre wyniki metodą jest w tym wypadku zastosowanie analizy językowej.

W najprostszym przypadku spośród wszystkich, wygenerowanych przez rozpoznawanie słów można wykluczyć te, które zawierają „podejrzane" sekwencje liter, np. „iy" czy też „iii". Ten rodzaj analizy językowej nie wymaga od programu przechowywania pełnej listy słów danego języka, a więc jest relatywnie mało „zasobochłonny" i szybki. Pamiętana jest jedynie częstość występowania zadanej sekwencji liter w konkretnym języku. Lepsze rezultaty daje algorytm, który „zna" pełną listę słów z danego języka i potrafi szybko ocenić, czy rozpoznane słowo znajduje się na tejże liście. Nawet dla języka polskiego, w którym pełny zbiór słów, wraz ze wszelkimi postaciami fleksyjnymi, liczy sobie ponad półtora miliona haseł (pięć razy więcej niż w języku angielskim), algorytmy analizy językowej potrafią skutecznie porównać z listą blisko 6000 jednostek leksykalnych na minutę (czytaj: „słów na minutę"), a cały materiał językowy mieści się w niespełna 2 MB pamięci RAM.

Są jednak sytuacje, w których analiza językowa, oparta wyłącznie na przeglądaniu słownika, nie daje poprawnych wyników. Przykładem może być wynik rozpoznawania podający jako warianty słowa „mila" i „miła". Każde z tych słów występuje w języku polskim, zatem o wyborze jednego z nich zadecydują prawdopodobieństwa rozpoznań poszczególnych znaków. W złej jakości wydrukach różnice między literą „l" a „ł" są tak nieznaczne, że algorytmy rozpoznawania przypiszą każdej z liter takie samo prawdopodobieństwo, równe 50%. Pomocna w tym wypadku mogłaby być analiza gramatyczna, która oceni, czy w rozpoznawanym zdaniu powinien pojawić się w danym miejscu rzeczownik „mila", czy też przymiotnik „miła".

Praktyczna implementacja analizy gramatycznej wymaga jednak znacznie większej ilości danych, a w szczególności konieczne jest zdefiniowanie i zakodowanie reguł gramatycznych obowiązujących w danym języku, np. za pomocą formalnego narzędzia typu HPSG (*Head-driven Phrase Structures Grammar*). O ile tego typu opisy, dla języków tzw. pozycyjnych, np. języka angielskiego, funkcjonują już w systemach komercyjnych, o tyle implementacja tego typu analizy dla języka polskiego (który jest językiem fleksyjnym) nie została do tej pory zrealizowana w jakimkolwiek komercyjnym ani też akademickim systemie ICR.

Cezary Grzegorz Dołęga

info

Grupa dyskusyjna

Pytania, uwagi i komentarze do artykułu można umieścić na liście dyskusyjnej news://news.vogel.pl/chip.software

Internet

FineReader Prof. 3.0:
http://www.mitcom.de/
OmniPage Pro 8.0:
http://www.caere.com/
Readiris 3.95:
http://www.irislink.com/US/us.html
Recognita Plus 4.0:
http://www.recognita.hu/
Textbridge Pro98:
http://www.textbridge.com/
Neurosoft Bip98:
http://www.neurosoft.com.pl/bip/