Grupp-projekt HT22, AI-Developer

Cat & Dog image classifier using Knn

Idén för programmet är att klassificera bilder på katter och hundar genom att använda algoritmen ”k-nearest-neighbour” och ”Euklides distans formel” , också kallad L2. Programmet kommer vara väldigt simpelt i sin natur, då vi använder oss av simpla metoder och formler för att göra beräkningarna, vilket av det jag läst inte realistiskt sett kan ge oss en särskilt hög träffsäkerhet. Jag hoppas på en träffsäkerhet högre än 60%.

Programmet och beräkningarna görs enbart på pixelvärdena i bilden. Tanken är här att det ska finnas ett visst mönster i de värden vi får ut av varje bild. Till exempel så borde vi kunna se en skillnad i värden mellan en hund och en katt, gissningsvis att en hund har ett större totalt värde än en katt. Det är alltså *ingen* riktig intelligens eller konkret inlärning som görs, då programmet lär sig ingenting från gång till gång eller från bild till bild, utan det är något vi själva ser. Programmet avgör enbart differensen mellan bilderna den testar mot, och i slutet görs en ”röstning” där k i knn är antalet värden vi ska ta med i röstningen, och i sin tur klassificera test-bilden om det är en hund eller en katt.

I praktiken betyder det om man utgår från exemplet ovan att har vi satt k = 7, så kommer programmet ge oss värdet på de 7 kortaste avstånden (”rösterna”) mellan test-bilden och träningsbilderna, där majoriteten av rösterna kommer avgöra om det är en katt eller hund.

På grund av hur ”dumt” programmet är, och ingen riktig inlärning görs, så kommer det alltså diffa väldigt mycket från bild till bild, då tex faktorer som bakgrunden på bilden, antalet objekt i bilden, storleken på katt/hund, päls-fluffighet och allt möjligt man kan komma på, kan öka eller minska pixelvärdet på bilden. Så tex på en bild där en människa är med och håller i en katt kommer värdet förmodligen bli mycket större än på en katt som sitter själv på golvet, vilket kan komma att ge den bilden etiketten ”hund”, fast det för oss uppenbart är en katt.

Programmet fungerar så att vi läser in alla tränings bilder (train) och modifierar dessa på de sätten vi valt (Färgskala, dimension) och sen konverterar bilden till en array för att få ut alla värden för varje enskild pixel i hela bilden. Det betyder att har vi dimensionerna 30x30 så blir det 900st pixlar, som alla har ett eget värde.

Värdet i det här fallet är färg-intensiteten (färg-styrkan) som pixeln har. I våra bilder har vi konverterat färgskalan till svartvit, vilket betyder att ett pixelvärde på 0 är helt svart, och ett värde på 255 är helt vitt.

Vi läser också in test-bilden/bilderna som vi gör samma procedur på.

Tanken är att i formeln L2 så är x1 bild 1, och x2 är bild 2. Det är alltså distansen mellan tränings-bild 1 och test-bild 1 som jämförs med varandra, och vi har i stunden jag skriver det här valt att summera alla pixlar i både tränings-bilden och test-bilden till ett värde för att enkelt kunna mäta distansen mellan hela bilderna. Det finns andra sätt att göra det på, tex att man tar delar av bilden och mäter mellan, men vi har inte testat det ännu. Varför vi valt hela bilden är för att det ger ett tydligt värde på differensen, man ser tydligt vad som skiljer bilderna åt samt om man visualiserar sig en graf så blir ju då varje bild en enskild punkt på grafen, istället för stora kluster av 900 pixlar per bild, som utgör den specifika bilden på grafen. Det kommer dock som tidigare nämnt med mindre träffsäkerhet då det är så många faktorer som bidrar med ”brus” på bilden, vilket avgör totala värdet på bilden.