**TIỂU LUẬN :**

**ỨNG DỤNG MẠNG NEURAL TRONG NHẬN DẠNG KÝ TỰ QUANG HỌC**

***GVHD : TS. Đỗ Phúc.***

***Sinh viên thực hiện :***

***Lương Vũ Trúc***

***Lâm Tú Bình***

***Huỳnh Thị Phương Oanh***

***Nguyễn Chí Thanh***

**Lời mở đầu**

Từ khi ra đời, máy tính đã nhanh chóng phát triển và đóng một vai trò rất quan trọng trong nghiên cứu khoa học kỹ thuật cũng như trong đời sống.

Nhưng một máy tính dù có mạnh đến đâu chăng nữa, cũng chỉ có thể làm việc theo một chương trình đã được hoạch định sẵn bởi lập trình viên.Nó vẫn không có khả năng liên tưởng, kết nối sự việc này với sự việc khác, và quan trọng hơn hết là khả năng sáng tạo như con người.

Vì lý do đó ,mạng noron (Artificial neural networks ) ra đời từ mục đích cố gắng mô phỏng hoạt động trí tuệ của con người.Từ khi ra đời, mạng neuron đã nhanh chóng phát triển trong các lĩnh vực về nhận dạng ,phân loại , giảm nhiễu, dự đoán…

Trong phạm vi một bài tiểu luận nhỏ, chúng em chỉ xin trình bày về “Ứng dụng mạng neuron trong nhận dạng ký tự quang học”.

Để hoàn thành bài tiểu luận này, chúng em đã được sự hướng dẫn nhiệt tình từ thầy TS Đỗ Phúc.Những bài giảng và tài liệu của thầy chính là cơ sở để chúng em có thể hoàn thành tốt báo cáo của mình. Chúng em xin chân thành cảm ơn thầy!

**Nhóm thực hiện**

**Chương 1 : *Tổng Quan Về Mạng Neural***

**I.Giới Thiệu**

Mạng neural nhân tạo (Artificial Neural Networks : ANN) ra đời xuất phát từ ý tưởng mô phỏng hoạt động của bộ não con người.

Mạng noron nhân tạo là sự tái tạo bằng kỹ thuật những chức năng của hệ thần kinh con người với vô số các neural được liên kết truyền thông với nhau qua mạng.Giống như con người , ANN được học bởi kinh nghiệm, lưu những kinh nghiệm đó và sử dụng trong những tình huống phù hợp.

Mạng neural trong một vài năm trở lại đây đã được nhiều người quan tâm và đã áp dụng thành công trong nhiều lĩnh vực khác nhau, như tài chính, y tế, địa chất và vật lý. Thật vậy, bất cứ ở đâu có vấn đề về dự báo, phân loại và điều khiển, mạng neural đều có thể ứng dụng được. Ví dụ như khả năng nhận dạng mặt người trong các hệ thống quản lý thông tin liên quan đến con người (quản lý nhân sự ở các công sở, doanh nghiệp; quản lý học sinh, sinh viên trong các trường trung học, đại học và cao đẳng;… ); các ngành khoa học hình sự, tội phạm; khoa học tướng số, tử vi,…

Kết hợp chặt chẽ với logic mờ, mạng neural nhân tạo đã tạo nên cuộc cách mạng thực sự trong việc thông minh hóa và vạn năng hóa các bộ điều khiển kỹ thuật cao cho cả hiện nay và trong tương lai. Ví dụ như ứng dụng tự động điều khiển hệ thống lái tàu, hệ thống dự báo sự cố,…

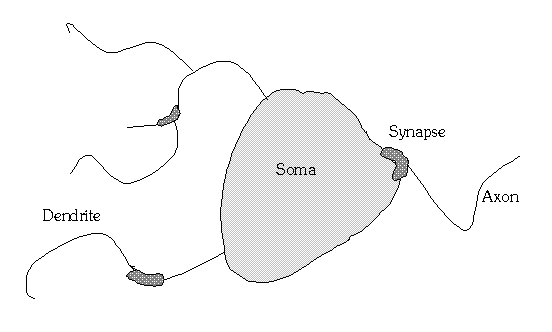
Mạng neural dựa trên việc mô phỏng cấp thấp hệ thống neural sinh học. Trong tương lai với sự phát triển mô phỏng neural sinh học, chúng ta có thể có loại máy tính thông minh thật sự.

*Trong phạm vi đề cập những gì liên quan đến kiến thức cơ sở để làm đồ án, chúng em chỉ xin trình bày những vấn đề có liên quan, những kiến thức được sử dụng phục vụ cho việc hoàn thành đồ án.*

**II.Khái Niệm Mạng Neural**

**1.Sơ lược về neural sinh học**

Sau đây là những thành phần chính trong cấu trúc của một nơron trong bộ não con người



Hình 1.1 : Mô hình neuron sinh học

Trong đó :

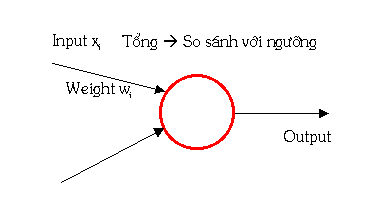
* Các Soma là thân của noron.
* Các dendrites là các dây mảnh, dài, gắn liền với soma, chúng truyền dữ liệu (dưới dạng xung điện thế) đến cho soma xử lý. Bên trong soma các dữ liệu đó được tổng hợp lại. Có thể xem gần đúng sự tổng hợp ấy như là một phép lấy tổng tất cả các dữ liệu mà nơron nhận được.
* Một loại dây dẫn tín hiệu khác cũng gắn với soma là các axon. Khác với dendrites, axons có khả năng phát các xung điện thế, chúng là các dây dẫn tín hiệu từ nơron đi các nơi khác. Chỉ khi nào điện thế trong soma vượt quá một giá trị ngưỡng nào đó (threshold) thì axon mới phát một xung điện thế, còn nếu không thì nó ở trạng thái nghỉ.
* Axon nối với các dendrites của các nơron khác thông qua những mối nối đặc biệt gọi là synapse. Khi điện thế của synapse tăng lên do các xung phát ra từ axon thì synapse sẽ nhả ra một số chất hoá học (neurotransmitters); các chất này mở "cửa" trên dendrites để cho các ions truyền qua. Chính dòng ions này làm thay đổi điện thế trên dendrites, tạo ra các xung dữ liệu lan truyền tới các nơron khác.
* Có thể tóm tắt hoạt động của một nơron như sau: nơron lấy tổng tất cả các điện thế vào mà nó nhận được, và phát ra một xung điện thế nếu tổng ấy lớn hơn một ngưỡng nào đó. Các nơron nối với nhau ở các synapses. Synapse được gọi là mạnh khi nó cho phép truyền dẫn dễ dàng tín hiệu qua các nơron khác. Ngược lại, một synapse yếu sẽ truyền dẫn tín hiệu rất khó khăn.

Các synapses đóng vai trò rất quan trọng trong sự học tập. Khi chúng ta học tập thì hoạt động của các synapses được tăng cường, tạo nên nhiều liên kết mạnh giữa các nơron.

Có thể nói rằng người nào học càng giỏi thì càng có nhiều synapses và các synapses ấy càng mạnh mẽ, hay nói cách khác, thì liên kết giữa các nơron càng nhiều, càng nhạy bén.

**2.Mạng Neural Nhân Tạo**

-Noron nhân tạo là một đơn vị tính toán có nhiều đầu vào và một đầu ra.

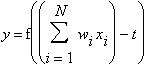


Hình 1.2 : Mô hình neuron nhân tạo ở mức đơn giản

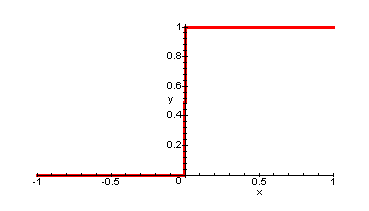
Nơron này sẽ hoạt động như sau: giả sử có N inputs, nơron sẽ có N weights (trọng số) tương ứng với N đường truyền inputs. Nơron sẽ lấy tổng cótrọng số của tất cả các inputs. Nói như thế có nghĩa là nơron sẽ lấy input thứ nhất, nhân với weight trên đường input thứ nhất, lấy input thứ hai nhân với weight của đường input thứ hai v.v..., rồi lấy tổng của tất cả các kết quả thu được. Đường truyền nào có weight càng lớn thì tín hiệu truyền qua đó càng lớn, như vậy có thể xem weight là đại lượng tương đương với synapse trong nơron sinh học. Có thể viết kết quả lấy tổng của nơron như sau:



Kết quả này sẽ được so sánh với threshold t của nơron, nếu nó lớn hơn t thì nơron cho output là 1, còn nếu nhỏ hơn thì output là 0. Ngoài ra ta cũng có thể trừ tổng nói trên cho t, rồi so sánh kết quả thu được với 0, nếu kết quả là dương thì nơron cho ouput bằng 1, nếu kết quả âm thì output là 0. Dưới dạng toán học ta có thể viết output của nơron như sau:

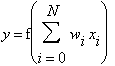


Trong đó f là hàm Heaviside:



f được gọi là threshold function hay transfer function của nơron, còn giá trị (-t) còn được gọi là bias hay offset của nơron.

Nếu chúng ta đưa thêm một input nữa vào, input thứ 0, có giá trị luôn luôn bằng 1 và weight luôn luôn bằng bias (-t) thì output của nơron còn có thể viết dưới dạng:



Lưu ý là chỉ số của tổng bây giờ bắt đầu từ 0 chứ không phải bằng 1 như trước nữa.

**3.Kiến Trúc Mạng**

Là một hệ thống bao gồm nhiều phần tử xử lý đơn giản (hay còn gọi là neural) tựa như neural thần kinh của não người, hoạt động song song và được nối với nhau bởi các liên kết neural. Mỗi liên kết kèm theo một trọng số nào đó, đặc trưng cho tính kích hoạt hoặc ức chế giữa các neural.

Có thể xem các trọng số là phương tiện để lưu trữ thông tin dài hạn trong mạng neural và nhiệm vụ của quá trình huấn luyện của mạng là cập nhật các trọng số khi có thêm thông tin về mẫu học. Hay nói một cách khác, các trọng số đều được điều chỉnh sao cho dáng điệu vào ra của mạng sẽ mô phỏng hoàn toàn phù hợp với môi trường đang xem xét.

**3.1 Mạng Một Tầng**

Mạng một tầng với S nơ-ron được minh họa trong hình 3.1. Chú ý rằng với mỗi một đầu vào trong số R đầu vào sẽ được nối với từng nơ-ron và ma trận trọng số bây giờ sẽ có S hàng.

Một tầng bao gồm ma trận trọng số, các bộ cộng, vector ngưỡng b, hàm chuyển và vector đầu ra a.

Mỗi phần tử của vector đầu vào **p** được nối với từng nơ-ron thông qua ma trận trọng số **W**. Mỗi nơ-ron có một ngưỡng , một bộ cộng, một hàm chuyển *f* và một đầu ra .



Cùng với nhau, các đầu ra tạo thành một vector đầu ra **a.**

Thông thường thì số lượng đầu vào của tầng khác với số lượng nơ-ron.(R#S).

Có phải tất cả các nơ-ron trong cùng một tầng phải có hàm chuyển giống nhau ? Câu trả lời là không; bạn có thể định nghĩa các nơ-ron trong một tầng có hàm chuyển khác nhau bằng cách kết hợp song song hai mạng nơ-ron giống ở trên. Cả hai sẽ có đầu vào giống nhau, và mỗi mạng sản xuất ra vài đầu ra.

Ma trận trọng số cho các phần tử trong vector đầu vào **W**:

**W =** 







*f*

*f*

*f*



**1**

**1**

**1**

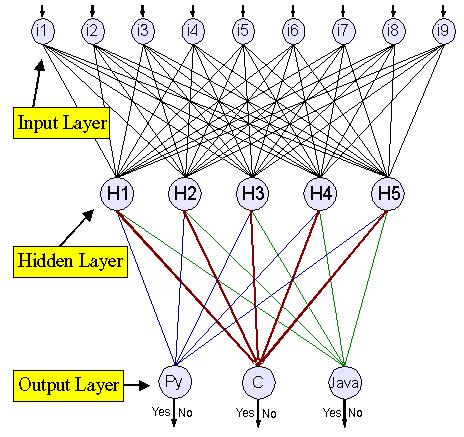
**a = f(Wp+b)**



Hình 3.1 : Mô hình mạng 1 lớp.

Các chỉ số hàng của các phần tử trong ma trận **W** chỉ ra nơ-ron đích đã kết hợp với trọng số đó, trong khi chỉ số cột cho biết đầu vào cho trọng số đó. Vì vậy, các chỉ số trong  nói rằng đây là trọng số của đầu vào thứ 2 nối với nơ-ron thứ 3.

**3.2 Mạng Đa Tầng**



Hình 3.2 :Ví dụ về một mô hình mạng đa tầng

Mô hình mạng neural ở trên gồm 3 lớp: lớp nhập (input), lớp ẩn(hidden) và lớp xuất (output). Mỗi nút trong lớp nhập nhận giá trị của một biến độc lập và chuyển vào mạng.

Dữ liệu từ tất cả các nút trong lớp nhập được tích hợp ­– ta gọi là tổng trọng số – và chuyển kết quả cho các nút trong lớp ẩn. Gọi là “ẩn” vì các nút trong lớp này chỉ liên lạc với các nút trong lớp nhập và lớp xuất, và chỉ có người thiết kế mạng mới biết lớp này (người sử dụng không biết lớp này).

Các nút trong lớp xuất nhận các tín hiệu tổng trọng hóa từ các nút trong lớp ẩn. Mỗi nút trong lớp xuất tương ứng với một biến phụ thuộc.

**III.Huấn luyện mạng neural.**

**1.Phương Pháp Học**

Mạng neural nhân tạo phỏng theo việc xử lý thông tin của bộ não người, do vậy đặc trưng cơ bản của mạng là có khả năng học, khả năng tái tạo các hình ảnh và dữ liệu khi đã học. Trong trạng thái học thông tin được lan truyền theo hai chiều nhiều lần để học các trọng số. Có 3 kiểu học chính, mỗi kiểu học tương ứng với một nhiệm vụ học trừu tượng. Đó là học có giám sát (có mẫu), học không giám sát và học tăng cường. Thông thường loại kiến trúc mạng nào cũng có thể dùng được cho các nhiệm vụ.

**1.1.Học có giám sát.**

Một thành phần không thể thiếu của phương pháp này là sự có mặt của một người thầy (ở bên ngoài hệ thống). Người thầy này có kiến thức về môi trường thể hiện qua một tập hợp các cặp đầu vào - đầu ra đã được biết trước. Hệ thống học (ở đây là mạng neural) sẽ phải tìm cách thay đổi các tham số bên trong của mình (các trọng số và các ngưỡng) để tạo nên một ánh xạ có khả năng ánh xạ các đầu vào thành các đầu ra mong muốn. Sự thay đổi này được tiến hành nhờ việc so sánh giữa đầu ra thực sự và đầu ra mong muốn.

**1.2.Học không giám sát.**

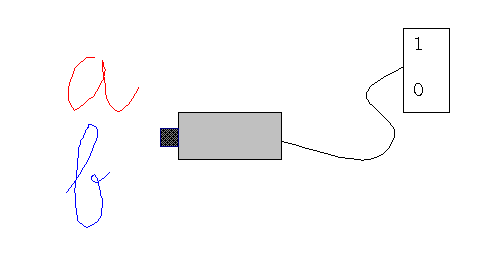
Trong học không có giám sát, ta được cho trước một số dữ liệu x và hàm chi phí cần được cực tiểu hóa có thể là một hàm bất kỳ của dữ liệu x và đầu ra của mạng, f – hàm chi phí được quyết định bởi phát biểu của bài toán. Phần lớn các ứng dụng nằm trong vùng của các bài toán ước lượng như mô hình hóa thống kê, nén, lọc, phân cụm.

**1.3.Học tăng cường.**

Dữ liệu x thường không được tạo trước mà được tạo ra trong quá trình một agent tương tác với môi trường. Tại mỗi thời điểm t, agent thực hiện hành động yt và môi trường tạo một quan sát xt với một chi phí tức thời Ct, theo một quy trình động nào đó (thường là không được biết). Mục tiêu là một sách lược lựa chọn hành động để cực tiểu hóa một chi phí dài hạn nào đó, nghĩa là chi phí tích lũy mong đợi. Quy trình hoạt động của môi trường và chi phí dài hạn cho mỗi sách lược thường không được biết, nhưng có thể ước lượng được. Mạng neural nhân tạo thường được dùng trong học tăng cường như một phần của thuật toán toàn cục. Các bài toán thường được giải quyết bằng học tăng cường là các bài toán điều khiển, trò chơi và các nhiệm vụ quyết định tuần tự (sequential decision making) khác.

**1.4.Ví dụ**

Giả sử chúng ta muốn dạy nơron phân biệt chữ A và B. Khi đưa input là A chúng ta muốn nơron cho output là 1, còn khi input là B thì nơron phải cho output bằng 0.



Hãy bắt đầu bằng cách cho các weights những giá trị ngẫu nhiên, lúc này nơron chưa biết gì hết. Bây giờ hãy input chữ A. Nơron sẽ lấy tổng có trọng số của các inputs và so sánh kết quả với 0. Nếu kết quả dương thì output là 1, âm thì output là 0. Khả năng nơron đoán đúng là 50%, vì các weights đang có giá trị hoàn toàn ngẫu nhiên. Nếu nơron đoán đúng thì chúng ta không cần làm gì cả, nhưng khi nơron đoán sai (output bằng 0), thì chúng ta phải tăng các weights của các inputs đang hoạt động (các inputs khác không) lên, sao cho lần tới tổng có trọng số sẽ vượt quá threshold và tạo nên output là 1.

Ngược lại, khi đưa chữ B vào và nơron đoán sai (output bằng 1), thì ta phải giảm các weights của các inputs đang hoạt động xuống, sao cho lần tới tổng có trọng số sẽ nhỏ hơn threshold và buộc nơron phải cho output bằng 0.

Như vậy, khi dạy chữ B thành công rồi thì nơron có quên đi chữ đã học trước đó là A không ? Không, vì khi input là các chữ khác nhau thì nhóm các đường inputs đang hoạt động cũng khác nhau hoặc là không hoàn toàn trùng nhau. Nhớ là chúng ta chỉ biến đổi weights của các inputs đang hoạt động thôi. Chúng ta chỉ việc lập đi lập lại quá trình dạy như trên cho tới khi nơron học thuộc bài mới thôi.

**2.Thuật Toán Huấn Luyện Mạng**

Trong phạm vi đề cập những gì liên quan đến kiến thức cơ sở để làm đồ án, chúng em chỉ xin trình bày những vấn đề có liên quan, những kiến thức được sử dụng phục vụ cho việc hoàn thành đồ án.

Trong đồ án , chúng em đã sử dụng mạng feed-forward và thuật toán lan truyền ngược sai số Back Propagation để xử lý bài toán.

**2.1.Thuật Toán Lan Truyền Ngược Back Propagation**

Đây là phương pháp thông dụng nhất để huấn luyện cho các mạng nơron truyền thẳng nhiều lớp.

Có thể áp dụng cho các mạng truyền thẳng với các hàm chuyển và các  
 hàm lỗi khả vi.

Tiêu chuẩn huấn luyện: Làm cho sai số đầu ra càng nhỏ càng tốt.

Triết lý của phương pháp: dựa trên đầu ra để điều chỉnh trọng số của lớp ra, sau đó dựa trên tính toán của lớp ra để điều chỉnh trọng số của lớp ẩn.

Huấn luyện mạng neural nhiều lớp sử dụng thuật toán Lan truyền ngược gồm hai quá trình: Quá trình truyền tuyến tính và quá trình truyền ngược:

Quá trình truyền tuyến tính: Dữ liệu từ lớp nhập qua lớp ẩn và đến lớp xuất để:

* Thay đổi giá trị của trọng số liên kết W của các neural trong mạng biểu diễn được dữ liệu học.
* Tìm ra sự khác nhau giữa giá trị thật hàm mẫu mà mạng tính được và kết quả dự đoán của mạng gọi là lỗi (học có giám sát).

Quá trình truyền ngược: Giá trị lỗi sẽ được truyền ngược lại sao cho quá trình huyến luyện(học) sẽ tìm ra trọng số Wi để lỗi nhỏ nhất.

*Công thức tính và áp dụng cụ thể cho bài toán sẽ được đề cập trong phần I.2 của Chương 2.*

#### Những hạn chế của phương pháp lan truyền ngược:

Ngoài những thành công của giải thuật học lan truyền ngược, vẫn còn có một số khía cạnh làm cho giải thuật trở nên chưa được bảo đảm là mọi lúc đều tốt. Khó khăn chủ yếu là ở quá trình huấn luyện lâu. Có thể do nhịp độ học và động lực không tối ưu. Sự sai sót trong việc huấn luyện nói chung xuất hiện từ hai nguồn: mạng liệt và những cực tiểu địa phương.

Mạng liệt: xảy ra khi những trọng số được điều chỉnh tới những giá trị rất lớn. Tổng đầu vào của một đơn vị ẩn hoặc đơn vị đầu ra có thể bởi vậy mà đạt giá trị rất cao (hoặc dương hoặc âm), và qua hàm kích hoạt sigmoid, đơn vị sẽ có một giá trị kích hoạt rất gần 0 hoặc rất gần 1. Giá trị hiệu chỉnh trọng số gần 0, và quá trình huấn luyện có thể đi đến một trạng thái dừng ảo.

Cực tiểu địa phương: bề mặt sai số của mạng rất phức tạp đầy những ngọn đồi và những thung lũng. Bởi vì sự hạ độ dốc, mạng có thể bị mắc bẫy tại một cực tiểu địa phương khi có nhiều cực tiểu thấp hơn gần bên cạnh. Những phương pháp thống kê có thể giúp để tránh cái bẫy này, nhưng chúng làm chậm. Một phương án khác là tăng thêm số lượng đơn vị ẩn. Như vậy sẽ làm việc trong không gian sai số nhiều chiều, nên cơ hội gặp bẫy nhỏ hơn. Tuy nhiên việc tăng cũng có giới hạn trên, khi vượt qua giới hạn này, cơ hội mắc bẫy lại tăng lên.

**Chương 2 : *Nhận Dạng Ký Tự Quang Học Trong Mạng Neural Nhân Tạo***

***Phương Pháp Và Thuật Toán Nhận Dạng Ký Tự***

**I.Các Bước Xây Dựng Nên Bài Toán**

Dưới đây là các bước chính khi tiến hành thiết kế và thực thi mạng noron nhân tạo cho bài toán nhận dạng ký tự :

Xác định dữ liệu đầu vào, và đầu ra mong muốn để tiến hành huấn luyện cho mạng noron.

* Đầu vào : Vì tiểu luận chỉ tập trung vào nghiên cứu và cài đặt quá trình nhận dạng ký tự neural, nên các dữ liệu đầu vào phải qua xử lý và đạt chuẩn yêu cầu trước khi đưa vào nhận dạng.Các yêu cầu dữ liệu đầu vào là các file ảnh dạng văn bản có định dạng \*.bmp, đây là dạng file ảnh chuẩn, có kiểu màu pixel đen và trắng, để việc tiến hành nhận dạng được chính xác.Các file ảnh này đã qua giai đoạn tiền xử lý : xử lý ảnh nhiễu, ảnh đa sắc , lọc ảnh...
* Đầu ra mong muốn : là một file text chứa các ký tự Unicode tương ứng mà chương trình mong muốn huấn luyện mạng học thuộc.Số ký tự đầu ra mong muốn phải tương ứng với số kí tự ảnh đầu vào cần huấn luyện.

ABCDEFGHIJKMNOPQRSTUVWXYZ

abcdefghijklmnopqrstuvwxyz

0123456789

+-\?\*^@&:<>

Hình 2.2 Một ví dụ về mẫu các ký tự trong nhận dạng ký tự quang học.

**2.Xác Định Cấu Trúc Mạng**

-Mạng noron được xây dựng theo phương pháp học có giám sát.

Bài toán lựa chọn mạng Feed-forward 3 lớp với cấu trúc như sau :

-Số noron lớp đầu vào : 150 noron.

Giải thích : Việc huấn luyện cho mạng học là một vòng lặp duyệt qua lần lượt các ký tự ảnh giúp mạng noron nhớ và nhận dạng các ký tự ảnh này.Với mỗi vòng lặp, một ký tự sẽ được đưa vào giảng dạy cho mạng noron học.Trong quá trình này, mỗi ký tự sẽ được phân tích và chuyển tỉ lệ vào một ma trận nhị phân 15 x 10 .Ứng với mỗi điểm lưới của ma trận sẽ được tuyến tính hóa tạo ra tương ứng với một noron đầu vào.Vì vậy ma trận sẽ tạo ra 15x10=150 noron đầu vào.

Lưu ý : Việc chọn số lượng noron đầu vào hay nói cách khác việc chọn tỉ lệ ma trận đầu vào cho kí tự là rất quan trọng.

**Nếu số lượng noron lớn quá sẽ dẫn đến việc :**

1.Nhận dạng những kí tự có font chữ nhỏ hoặc trung bình bị sai hoặc không chính xác.

2.Chương trình viết ra phải duyệt qua rất nhiều vòng lặp dẫn đến chương trình bị chậm, hoặc gây ra tình trạng bị đứng máy khi chạy.

**Nếu số lượng noron quá nhỏ :**

-Việc phân tích các kí tự ảnh thông thường và lớn sẽ dẫn đến sai số lớn , vì vậy, việc nhận dạng kí tự cũng sẽ thiếu tính chính xác.

-Qua kiểm nghiệm, và dựa trên nhiều tài liệu, nhận thấy giá trị 150 noron đầu vào là khá phù hợp để nhận dạng với kiểu kí tự ảnh có font chữ trung bình, và phổ biến.Các kí tự ảnh này có chiều cao trung bình là 70 pixel và chiều rộng khoảng 54 pixel.

-Số noron lớp ẩn : 500 noron.Việc lựa chọn giá trị này dựa trên thực nghiệm của các tài liệu chuyên về nhận dạng kí tự quang học.Số noron này được quyết định từ những kết quả tối ưu và giá trị lỗi cơ sở trong quá trình huấn luyện.

-Số noron đầu ra : 16 noron tương ứng với 16 bit nhị phân của mã Unicode.Từ 16 noron này ta tổng hợp và chuyển thành 16 bit nhị phân, sau cùng chuyển thành 1 kí tự Unicode.

500 neural Lớp ẩn

150 neural Lớp đầu vào

**Điểm ảnh**

**Điểm ảnh**

**Điểm ảnh**

**Điểm ảnh**

**0 hoặc 1**

**0 hoặc 1**

**0 hoặc 1**

**0 hoặc 1**

16 neural Lớp đầu ra

**3.Thuật toán huấn luyện mạng.**

Mạng feed- forward sử dụng giải thuật lan truyền ngược sai số Back Propagation.

Giải thuật gồm 2 bước :

* **Bước 1:**

-Lan truyền xuôi đầu vào qua mạng.

Sử dụng các công thức :

Công thức chung tính đầu ra của một noron thứ i tại lớp thứ k:

yi= f(-bi)

Với f là hàm chuyển sigmoid lưỡng cực được tính theo công thức :

f=

α: hệ số góc của hàm chuyển

t: biến net-input

bi: hệ số ngưỡng hay độ lệch

**Áp dụng đối với mô hình mạng của chương trình :**

-Công thức cho đầu ra của một noron thứ i (1🡪250) tại lớp ẩn

ai= f(-bi)

Với : trọng số tại noron thứ i của lớp ẩn kết nối với đầu vào thứ j của lớp vào

: giá trị đầu vào của noron thứ j tại lớp vào

bi : giá trị ngưỡng hay độ lệch của noron thứ i của đầu vào

-Công thức cho đầu ra của một noron thứ k (k =1🡪16)tại lớp output

yk= f()=f [f()]

* **Bước 2** : Lan truyền ngược

Tính toán sai lệch giữa đầu ra thực và đầu ra mong muốn của noron thứ k tại đầu ra.

ei= ti- yi

Tổng bình phương sai số của mạng ứng với mẫu học (Xs, Ts):

E=( ti- yi)2

P: số phần tử đầu ra

Nếu E< ε và l+1<epochs thì :

Thông tin sai số sẽ được lan truyền ngược qua mạng để điều chỉnh lại trọng số tại vòng lặp l.

-Công thức điều chỉnh trọng số với liên kết giữa noron thứ j trong lớp ẩn và noron thứ i trong lớp ra tại lần lặp l+1: (l+1<số lần dạy( epochs) )

(l+1) = (l)+η.ei(l).yj.f’(yi(l))

Với η: hệ số học

ei(l) : giá trị sai lệch của noron thứ i trong lớp ra , trong lần dạy (lặp ) thứ l .

f’: đạo hàm của hàm chuyển lưỡng cực ,công thức f’=

 η : hệ số học

yi(l) : giá trị đầu ra của noron thứ i trong lớp ra tại vòng lặp thứ l.

yj (l): giá trị đầu ra của noron thứ j trong lớp ẩn tại vòng lặp thứ l.

-Công thức điều chỉnh trọng số với liên kết giữa noron vào thứ j và noron ẩn thứ i , tại lần lặp thứ l+1 (l+1 <epochs)

(l+1) = (l)+ η.xj.f’(yi).

Với :

η: hệ số học

xj : giá trị đầu ra của noron thứ j trong lớp vào.

yi : giá trị đầu ra của noron thứ i trong lớp ẩn

wki(l+1) : trọng số liên kết giữa noron thứ k trong lớp ra và noron thứ i trong lớp ẩn trong lần lặp thứ l+1 .

yk (l) : giá trị đầu ra của noron thứ k trong lớp ra.

* **Các tham số sử dụng trong chương trình**

Tốc độ học η = 150.

Hệ số góc α Sigmoid= 0.014. 

Giá trị ngưỡng hay độ lệch : 30

Số lần dạy 300-600 tùy độ phức tạp của từng loại font.

Ngưỡng của lỗi = 0.0002.

**II.Phương Pháp và Thuật Toán Nhận Dạng Kí Tự**

1**.Xử lý dữ liệu (Phân tích ảnh)**

-Quá trình phân tích ảnh thực chất là phân tích chuỗi văn bản ảnh thành từng ký tự một. Việc phân tích này dựa trên việc nhận dạng các pixel màu của ký tự ảnh với dữ liệu đầu vào đã được xử lý và chuyển thành định dạng \*.bmp chỉ có 2 pixel điểm đen RGB(0,0,0) và điểm trắng RGB(255, 255, 255).

Quá trình phân tích ảnh được thực hiện tuần tự như sau :

-Phân tích văn bản thành từng dòng văn bản .

-Phân tích từng dòng văn bản thành từng ký tự ảnh một.

**1.1.Tách dòng kí tự**

- Việc tách dòng kí tự chính là cơ sở để tiến hành tách từng kí tự một cách dễ dàng và nhanh chóng.

* Thuật toán :

-Bắt đầu từ điểm ảnh (x, y) đầu tiên của file ảnh (0,0) , gán số dòng đang xét lines =0;

a.-Xác định giới hạn trên của dòng : Quét hết chiều rộng của ảnh, trên cùng một giá trị y ( chạy x<=chiều rộng ảnh, giữ nguyên y )

+ Nếu gặp một pixel đen thì ghi nhận y chính là tọa độ giới hạn trên của dòng(top\_line).Dừng duyệt x.

+Nếu không , tiếp tục duyệt

+Nếu khi quét hết chiều rộng của ảnh, vẫn không tìm thấy giá trị điểm màu đen thì reset lại x=0, và tiếp tục duyệt đến dòng tiếp theo (tăng y=y+1)

b.-Xác định giới hạn dưới của dòng :

+ Bắt đầu duyệt từ giới hạn trên (đỉnh ) vừa tìm thấy của dòng (0,top\_line)

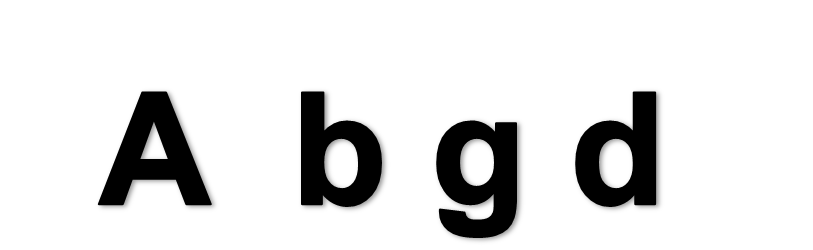
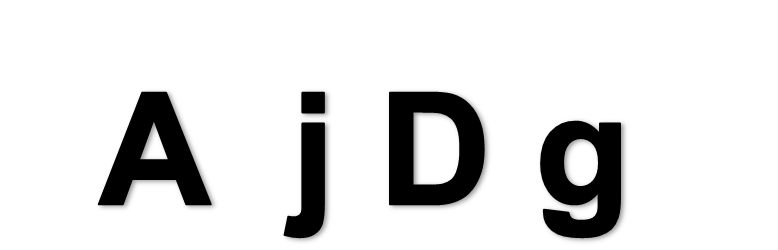
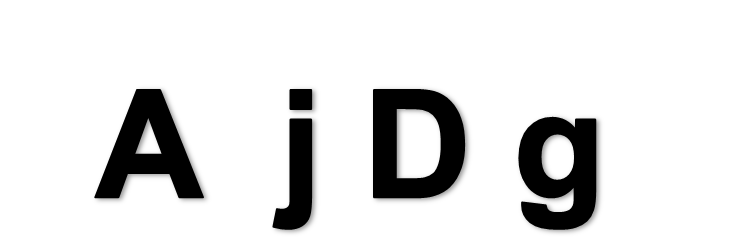
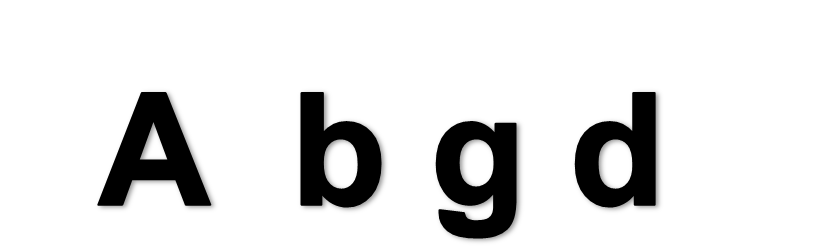
+ Tương tự như xác định giới hạn trên , ta duyệt hết chiều rộng của ảnh trên cùng một giá trị y.

+Nếu duyệt hết dòng mà không tìm thấy ký tự pixel đen nào thì ghi nhận y-1 là giới hạn dưới của dòng (bottom\_line).Dừng duyệt.Tăng số dòng lên (lines++).

+Nếu chưa tìm thấy bottom\_line, tiếp tục duyệt đến dòng tiếp theo (tăng y, reset x=0).

c.Bắt đầu từ giới hạn dưới y (bottom\_line) vừa tìm thấy sau cùng,lặp lại các bước a,b để xác định các giới hạn của các dòng tiếp theo , cho đến khi duyệt hết chiều cao của ảnh thì dừng, quá trình xác định dòng ký tự hoàn tất.

*Ảnh đầu vào*



Quá trình tách dòng

**Giới hạn trên**

**Giới hạn dưới**

**1.2.Tách từng kí tự**

* Thuật toán :

Xét lần lượt từng dòng

1.-Bắt đầu xét từ giá trị giới hạn trên y của dòng (top\_line) và giá trị x đầu tiên (x=0)

2.Xác định giới hạn trên của ký tự : Quét hết chiều rộng của ảnh, trên cùng giá trị y.

+ Nếu phát hiện pixel đen thì đánh dấu y là giá trị đỉnh của ký tự (top\_character).Dừng quét.

+Nếu quét hết chiều rộng, mà vẫn không tìm thấy pixel đen nào thì tăng y và reset lại x, tiếp tục thực hiện lại bước 2.

3. Xác định giới hạn dưới của ký tự :Bắt đầu duyệt từ giới hạn trên (đỉnh ) vừa tìm thấy của ký tự (0,top\_character)

+ Tương tự như xác định giới hạn trên , ta duyệt hết chiều rộng của ảnh trên cùng một giá trị y.

+Nếu duyệt hết dòng mà không tìm thấy ký tự pixel đen nào thì ghi nhận y-1 là giới hạn dưới của ký tự (bottom\_character).Dừng duyệt.

+Nếu chưa tìm thấy bottom\_character, tiếp tục duyệt đến dòng tiếp theo (tăng y, reset x=0).

4.Xác định giới hạn trái của ký tự (xác định giá trị của x )

-Bắt đầu từ giới hạn trên (đỉnh của ký tự - top\_character), giá trị x đầu tiên (x=0) .

-Quét đến giới hạn dưới của dòng (bottom\_character), giữ nguyên x (quét theo chiều thẳng đứng )

+Nếu gặp pixel đen đầu tiên, ghi nhận x là giới hạn trái của kí tự (left-character).Dừng quét.

+Nếu quét đến cuối giới hạn dưới, vẫn không tìm thấy pixel đen nào, thì reset lại y = giới hạn trên vừa tìm thấy, tăng x lên.(x++),và tiếp tục thực hiện lại bước 4.

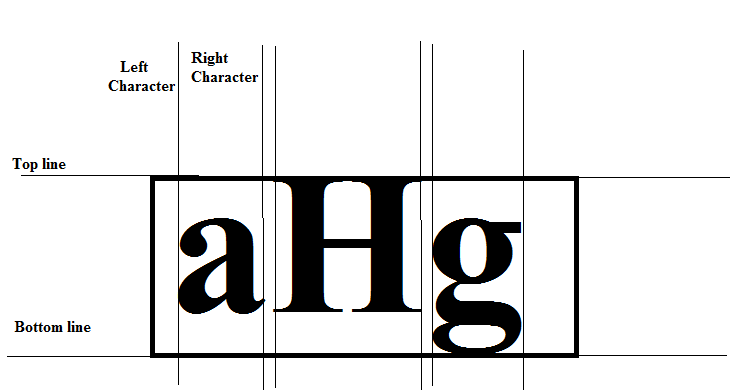
5.Xác định giới hạn phải của ký tự (xác định giá trị x )

-Bắt đầu từ giới hạn trên -đỉnh của ký tự (top\_character), giới hạn trái của ký tự (left\_character,top\_character) .Quét theo chiều thẳng đứng đến giới hạn dưới của dòng.

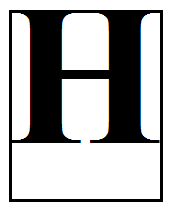
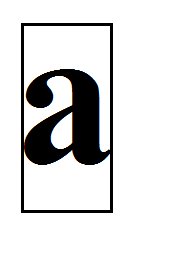
+Nếu khi quét hết chiều đứng ,mà vẫn không gặp được pixel đen nào thì ghi nhận x-1 là giới hạn phải của ký tự (right\_character).Dừng quét.

+Nếu gặp pixel đen thì tăng x (x++) và reset lại y =top\_character của ký tự đang xét ,để xét đường thẳng đứng tiếp theo.

6. Lặp lại bước 1 đến 5 để xác định giới hạn của ký tự tiếp theo trên cùng dòng.Với y=top\_line của dòng đang xét và giá trị x đầu tiên = right\_character của ký tự vừa tìm thấy.



Sau khi tách ký tự



**1.3.Xác Định Giới Hạn Chính Xác Cho Từng Ký Tự**

Trong một số trường hợp, việc tách ký tự chưa thực sự chính xác (giống như ví dụ trên ).Ta cần xác định lại một lần nữa giới hạn chính xác cho từng ký tự.

* **Thuật toán :**

Xét từng ký tự :

***1.Xác định đỉnh ký tự***

-Bắt đầu từ đỉnh của ký tự tạm thời,và giới hạn trái của ký tự vừa xác định ở trên( left\_character,top\_character).Quét đến giới hạn phải của ký tự, giữ nguyên y.

+Nếu tìm thấy pixel đen thì chọn lại y là giới hạn đỉnh của ký tự (top\_character=y).Dừng quét.

+Nếu không , xét điểm tiếp theo.

+Nếu không tìm thấy pixel đen nào thì tăng y (y++) , reset lại x .(x= left\_character).Thực hiện lại bước 1.

***2.Xác định giới hạn dưới của ký tự.***

- Bắt đầu từ giới hạn dưới hiện thời của ký tự (được xác định ở trên –phần 1.2 ), và giới hạn trái của ký tự (left\_character, bottom\_character).Quét x đến giới hạn phải của ký tự, trên cùng giá trị y .

+ Nếu tìm thấy pixel đen ,đánh dấu y là giới hạn dưới của ký tự ( bottom\_character= y).Ngừng quét.

+Nếu không tìm thấy pixel đen nào, thì reset lại x (= left character ), giảm y (y--).Thực hiện lại bước 2.

Xác định giới hạn

Giới hạn trên

Giới hạn dưới

**1.4.Ánh Xạ Ký Tự Ảnh Vào Ma Trận Giá Trị**

Sau khi tách, ký tự ảnh sẽ được ánh xạ vào ma trận 2 chiều, có kích thước 15 x 10. Vì các ký tự ảnh dù đã chuẩn hóa, chọn lọc nhưng sau khi tách đều có kích thước khá lớn so với ma trận ( khoảng 58x 74), nên các ký tự ảnh này đều phải qua 1 bước chuyển đổi tỉ lệ trước khi chuyển giá trị vào ma trận nhị phân 15 x10.

* Lưu ý : Việc chuyển đổi tỉ lệ này là một trong những nguyên nhân gây ra sai số dẫn đến nhận dạng ký tự bị sai.

**Thuật toán** : ánh xạ ký tự ảnh vào ma trận pixel ảnh.

***a.Xét chiều rộng***

-Ánh xạ điểm đầu và điểm cuối của ký tự ảnh tương ứng với giá trị đầu và cuối của ma trận .

-Khởi tạo với 10 phần tử tương ứng.

-Ánh xạ tọa độ điểm đầu (0,y) và điểm cuối (width,y) của ảnh kí tự tương ứng với giá trị đầu (0,y) và giá trị cuối (10,y) của ma trận.

-Chia nhỏ chiều rộng thành 10 mốc giá trị, ánh xạ vào chiều rộng của ma trận pixel.

Ví dụ :

xMT[ 0]=0;

xMT[5]=width/2.

xMT[10]=width.

xMT[2]=xMT[5]/2.

.....

***b.Đối với chiều cao:***

Khởi tạo với 15 phần tử tương ứng.

Ánh xạ điểm đầu (x,0) và điểm cuối (x,height) của ảnh kí tự tương ứng với giá trị đầu (x,0) và giá trị cuối (x,15) của ma trận.

Chia nhỏ chiều cao thành 15 mốc giá trị, ánh xạ tương ứng vào chiều cao của ma trận pixel.

Vd: yMT[ 0]=0;

yMT[5]=height /3.

xMT[10]=height x 2/3.

xMT[15]=height.

...

-Lấy các giá trị pixel trong file ảnh tuyến tính hóa vào trong ma trận pixel bằng cách kết hợp tọa độ chiều cao và chiều rộng.

Vd:MT[i,j]=img[xMT[i],yMT[j] ]



Hình : Ánh xạ lưới ký tự.

***c.Chuyển ma trận lưới pixel thành ma trận giá trị***

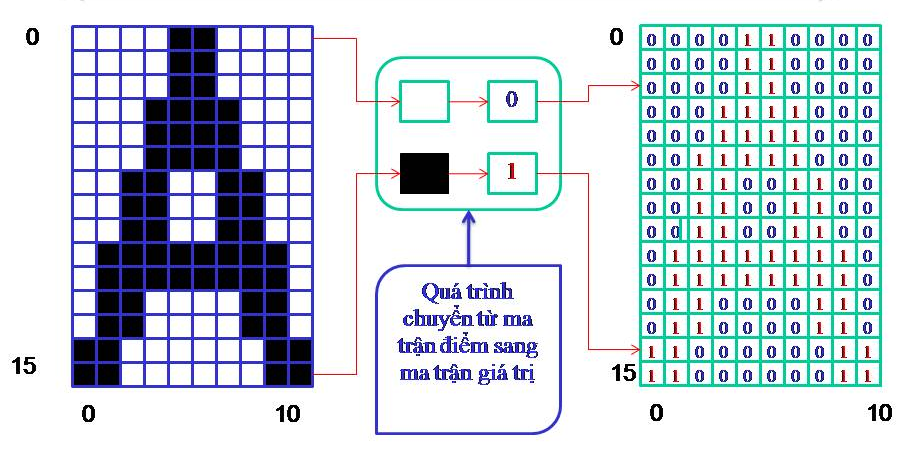
-Bắt đầu từ tọa độ (0,0) của ma trận pixel.Giữ y , duyệt qua x:

Nếu pixel đen thì ghi nhận giá trị 1 vào ma trận giá trị có tọa độ tương ứng.

Nếu pixel trắng thì ghi nhận giá trị 0 vào ma trận giá trị có tọa độ tương ứng.

Nếu x= chiều rộng thì reset lại x=0 , tăng y (y++).

***Từ ma trận ảnh này ta ánh xạ thành mảng 1 chiều 150 cột 1 dòng, và chuyển thành 150 noron đầu vào.***



**2.Huấn Luyện Mạng**

* **Thuật toán :**

1.Xây dựng cấu trúc mạng với mô hình 3 lớp .

2.Khởi tạo các giá trị ngẫu nhiên cho trọng số trong khoảng [-độ lệch,độ lệch ]

3.Nạp file ảnh đầu vào và file huấn luyện (file text đầu ra mong muốn ).

4.Phân tích ảnh và ánh xạ thành 150 noron đầu vào.

5.Đọc các kí tự đầu ra mong muốn, chuyển thành giá trị nhị phân Unicode và lưu trữ lại.

6.Với mỗi kí tự ảnh :

+Tính giá trị đầu ra trong mạng FeedForward.

+So sánh với ký tự tương ứng ở đầu ra mong muốn và tính toán lỗi.

+Lan truyền ngược lỗi và điều chỉnh lại giá trị của trọng số .

7.Chuyển đến xét ký tự ảnh tiếp theo thực hiện lại bước 6 , cho đến khi duyệt hết các ký tự ảnh.

8.Tính giá trị lỗi trung bình cho tất cả các ký tự ảnh.

9. Lặp lại bước 6 đến bước 8 cho đến khi chạy đủ số vòng dạy

+Nếu lỗi trung bình đạt thấp hơn giá trị ngưỡng của lỗi thì dừng .

+Nếu không thì tiếp tục.

Xây dựng mạng

Khởi tạo trọng số

Nạp file huấn luyện

Phân tích ảnh

Phát hiện kí tự tiếp theo

Các kí tự sẵn sàng ?

Các kí tự sẵn sàng ?

Đọc đầu ra mong muốn

Tính đầu ra của mạng

Tính lỗi

Các kí tự sẵn sàng ?

Lỗi <Ngưỡng ?

Số lần lặp =max epochs ?

Tính trung bình lỗi

Vector đầu vào tiếp theo

Đ

S

**3.Nhận Dạng Ký Tự Quang Học**

* Thuật toán :

-Nạp file ảnh đầu vào

-Phân tích ảnh thành từng ký tự ảnh.

-Đưa ma trận giá trị cho đầu vào mạng feed-forward , tính toán giá trị đầu ra dựa trên các trọng số đã được lưu sẵn khi tiến hành huấn luyện mạng.

-Chuyển mã nhị phân Unicode thành ký tự tương ứng.

Đ

S

Đ

S

Tính toán đầu ra

Dòng đầu tiên

Ánh xạ kí tự tới vector

Phân tích dòng ảnh

Chuyển từ Unicode sang kí tự

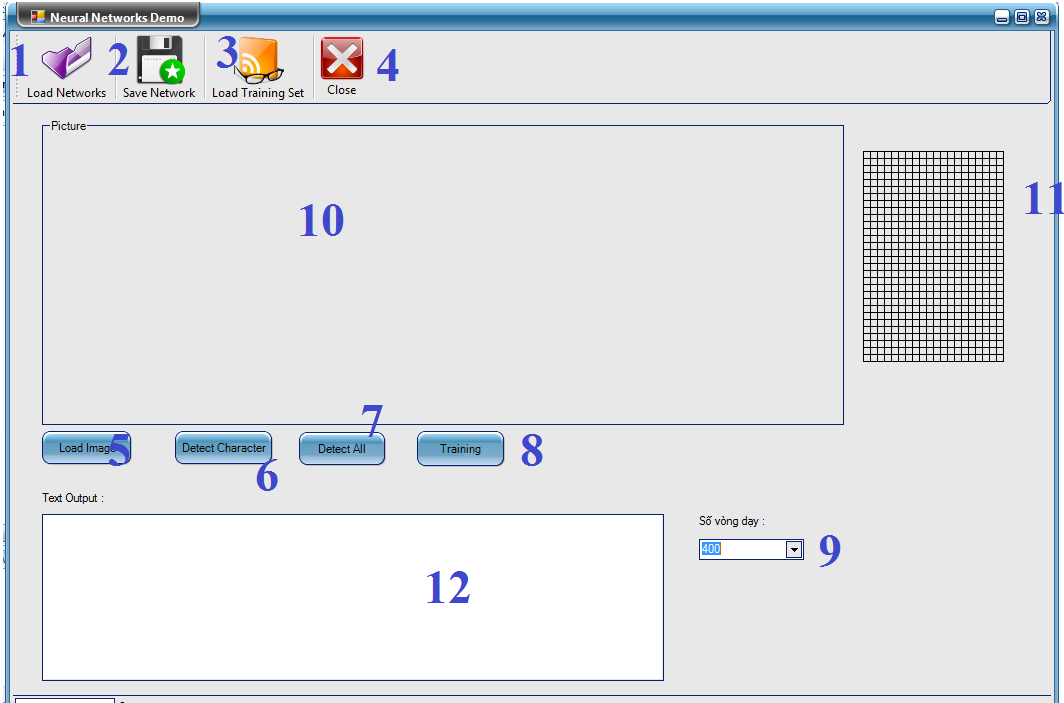
Hết kí tự trên dòng ?

Hết Các dòng ?

Dòng tiếp theo

Kí tự tiếp theo

**Chương 3 :*Giao Diện Chính Chương Trình***



* Chú Thích

1.Load Networks : load các trọng số đã được huấn luyện lên để phục vụ cho việc nhận dạng kí tự.

2.Save Networks : save network đã được huấn luyện lại sử dụng cho những lần đọc dữ liệu sau.

3.Load Training Set : load tập dữ liệu mẫu muốn huấn luyện lên.Tập dữ liệu mẫu này gồm 2 file cùng tên khác định dạng (\*.ts) và (\*.nn).

4.Close : Đóng chương trình.

5.Load Image : đưa ảnh đầu vào vào để phân tích.

6.Detect Character : Duyệt từng kí tự một.

7.Duyệt qua tất cả các kí tự.

8.Trainning : trainning mạng,dùng khi muốn load trainning set vào để dạy cho mạng.

9.Số vòng dạy : tùy vào độ phức tạp của font chữ, và sai số có thể điều chỉnh số vòng dạy phù hợp.

10.Khung hiện ảnh đưa vào.

11. Ma trận 20 x30 thể hiện ảnh pixel các kí tự sau khi ánh xạ.

12.Khung hiện nội dung chữ sau khi nhận dạng.

* **Hướng dẫn :**
* Để nhận dạng kí tự bạn có thể load mạng đã được huấn luyện sẵn lên (load network) , rồi chọn ảnh muốn nhận dạng (load image ) và tiến hành nhận dạng ảnh (detect character hoặc detect all).
* Hoặc load tập dữ liệu mẫu vào , tiến hành trainning , rồi mới tiến hành load ảnh muốn nhận dạng kí tự.



***Tổng Kết***

* ***Sai Số Trong Nhận Dạng Ký Tự***

Việc nhận dạng ký tự đôi khi bị sai hoặc thiếu chính xác , là do nhiều vấn đề liên quan như : sự phức tạp của dữ liệu đầu vào, các giá trị chọn lựa như độ lệch, lỗi ngưỡng, số neuron mỗi lớp,…. chưa được chọn lựa chính xác.

* Sai số trong nhận dạng kí tự lớn hay nhỏ là phụ thuộc vào nhiều yếu tố : độ chuẩn của ảnh đưa vào, size font chữ , độ phức tạp của font chữ.
* Việc chọn mô hình mạng phù hợp, giá trị ngưỡng lỗi, độ lệch , tốc độ học.. đều ảnh hưởng đến tính chính xác trong nhận dạng kí tự
* ***Những hạn chế***

+ Chương trình chỉ ở mức nhận dạng chữ không dấu , tính chính xác vẫn chưa hoàn toàn tuyệt đối.

+Với những kiểu chữ sát nhau, việc phân tích giới hạn ký tự chưa chính xác, dẫn đến đọc ký tự sai.

Vd :



* Do kiến thức còn hạn hẹp, thời gian làm báo cáo cũng không nhiều, nên phần báo cáo của chúng em chắc chắn còn rất nhiều sai sót .Mong thầy thông cảm và bỏ qua cho chúng em.Chúng em xin cảm ơn
* ***Nhóm thực hiện*** : nhóm 12.Gồm các thành viên :
* Lâm Tú Bình -07520427
* Lương Vũ Trúc -07520376
* Huỳnh Thị Phương Oanh -07520263
* Nguyễn Chí Thanh -07520312

**TÀI LIỆU THAM KHẢO**

[1.] Các bài giảng về neural networks trên web.

[2.] http://vi.wikipedia.org

[3.] Tạp chí khoa học và công nghệ,Đại học Đà Nẵng – Số 4(27).2008

[4.] An Introduction to Neural Networks.pdf

[5.] http:/ /codeproject.com

[6.]Và một số nguồn trên http://google.com.