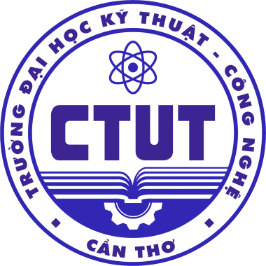
********

**Sinh viên:** **Tạ Phạm Phú Điền MSSV: 1700542**

**Lê Tấn Phát MSSV: 1700677**

Trường Đại học Kỹ thuật - Công nghệ Cần Thơ

**ĐỀ TÀI: TÌM HIỂU VỀ MẠNG NEURON**

**VÀ ỨNG DỤNG MINH HỌA**

BÀI BÁO CÁO

TRÍ TUỆ NHÂN TẠO

**Mục lục**

[**LỜI NÓI ĐẦU** 3](#_Toc5879379)

[1.TỔNG QUAN MẠNG NEURON 4](#_Toc5879380)

[1.1 MẠNG NEURON NHÂN TẠO 4](#_Toc5879381)

[1.2 ỨNG DỤNG MẠNG NEURON NHÂN TẠO 4](#_Toc5879382)

[**2. CẤU TRÚC MẠNG NEURON** 4](#_Toc5879383)

[**3.HỌC CÓ GIÁM SÁT VÀ HỌC KHÔNG CÓ GIÁM SÁT** 5](#_Toc5879384)

[3.1 MẠNG NEURON HỌC CÓ GIÁM SÁT 5](#_Toc5879385)

[3.2 MẠNG NEURON HỌC KHÔNG CÓ GIÁM SÁT 5](#_Toc5879386)

[3.3 MẠNG NEURON HỌC TĂNG CƯỜNG 5](#_Toc5879387)

[**4. GIẢI THUẬT LAN TRUYỀN NGƯỢC** 5](#_Toc5879388)

[**5. LEARNING RATE** 6](#_Toc5879389)

[**6. MỘT SỐ KIỂU MẠNG NEURON NHÂN TẠO** 6](#_Toc5879390)

[6.1 Mạng tự tổ chức (Kohonen feature maps) 6](#_Toc5879391)

[6.2 Mạng nơ ron truyền thẳng đa tầng (Multilayer Perceptrons). 7](#_Toc5879392)

[**7. THIẾT KẾ MẠNG NEURON** 8](#_Toc5879393)

[7.1Số lượng lớp ẩn (trong mạng Backpropagation): 8](#_Toc5879394)

[7.2 Mạng neuron ban đầu 8](#_Toc5879395)

[7.3 Số mẫu huấn luyện 8](#_Toc5879396)

[**8. CÀI ĐẶT MÔI TRƯỜNG PHÁT TRIỂN** 8](#_Toc5879397)

# **LỜI NÓI ĐẦU**

Trí tuệ nhân tạo là một môn khoa học và một tập hợp các công nghệ máy  
tính được lấy cảm hứng từ - nhưng hoạt động theo cách hoàn toàn khác - cách con  
người sử dụng hệ thống thần kinh và cơ thể của mình để cảm nhận, học hỏi, lý  
giải, và hành động. Mặc dù các tiến bộ trong trí tuệ nhân tạo chắp vá và không thể  
đoán trước, nhưng lĩnh vực này đã có những bước tiến đầy ý nghĩa kể từ khi bắt  
đầu sáu mươi năm trước.

Chúng ta hiện đang bước vào kỷ nguyên trong đó trí tuệ nhân tạo có những  
tác động to lớn và sâu sắc đến đời sống hàng ngày. Thí dụ, thị giác máy tính và trí  
tuệ nhân tạo lập kế hoạch tạo ra các trò chơi video giờ đây trở thành một ngành  
công nghiệp giải trí lớn hơn Hollywood. Học sâu, một hình thức học máy dựa trên  
các lớp đại diện của biến số được xem như các mạng như thần kinh, đã làm cho  
việc hiểu lời nói trở thành thực tế trên điện thoại và trong nhà bếp của chúng ta.  
Xử lý ngôn ngữ tự nhiên cùng với lý giải và thể hiện tri thức đã cho phép một máy  
tính đánh bại nhà vô địch Jeopardy và đang mang lại sức mạnh mới cho tìm kiếm  
Web.

Kế hoạch chiến lược quốc gia nghiên cứu và phát triển trí tuệ nhân tạo đưa ra  
một khuôn khổ cấp cao có thể sử dụng để xác định các nhu cầu khoa học và công nghệ trong trí tuệ nhân tạo, và theo dõi tiến bộ và tối đa hóa tác động của các đầu tư NC&PT vào các nhu cầu đó. Đồng thời Kế hoạch cũng đặt ra các ưu tiên nghiên cứu được chính phủ đầu tư, xem xét các khả năng của trí tuệ nhân tạo có các tác động chuyển hóa lâu dài lên xã hội và thế giới.

## **1.TỔNG QUAN MẠNG NEURON**

## **1.1 MẠNG NEURON NHÂN TẠO**

**Mạng neuron nhân tạo** hay thường gọi ngắn gọn là [**mạng**](https://vi.wikipedia.org/wiki/M%E1%BA%A1ng_n%C6%A1-ron) **neuron** là một [mô hình toán học](https://vi.wikipedia.org/wiki/M%C3%B4_h%C3%ACnh_to%C3%A1n_h%E1%BB%8Dc) hay [mô hình tính toán](https://vi.wikipedia.org/wiki/M%C3%B4_h%C3%ACnh_t%C3%ADnh_to%C3%A1n) được xây dựng dựa trên các [mạng nơ-ron sinh học](https://vi.wikipedia.org/w/index.php?title=M%E1%BA%A1ng_n%C6%A1-ron_sinh_h%E1%BB%8Dc&action=edit&redlink=1). Nó gồm có một nhóm các [nơ-ron nhân tạo](https://vi.wikipedia.org/w/index.php?title=N%C6%A1-ron_nh%C3%A2n_t%E1%BA%A1o&action=edit&redlink=1) (nút) nối với nhau, và xử lý thông tin bằng cách truyền theo các kết nối và tính giá trị mới tại các nút (cách tiếp cận connectionism đối với [tính toán](https://vi.wikipedia.org/wiki/T%C3%ADnh_to%C3%A1n)). Trong nhiều trường hợp, mạng nơ-ron nhân tạo là một [hệ thống thích ứng](https://vi.wikipedia.org/w/index.php?title=H%E1%BB%87_th%E1%BB%91ng_th%C3%ADch_%E1%BB%A9ng&action=edit&redlink=1) (*adaptive system*) tự thay đổi cấu trúc của mình dựa trên các thông tin bên ngoài hay bên trong chảy qua mạng trong quá trình học.

## **1.2 ỨNG DỤNG MẠNG NEURON NHÂN TẠO**

Một mạng neuron nhân tạo được cấu hình cho một ứng dụng cụ thể (nhận dạng mẫu,phân loại dữ liệu,…) thông qua một quá trình học từ các mẫu huấn luyện. Về bản chất học chính là quá trình hiệu chỉnh trọng số liên kết giữa các neuron

**2. CẤU TRÚC MẠNG NEURON**



**Giải thích ký hiệu:**

• Tập các đầu Vào: là các tín hiệu vào (input signal) của neuron, các tín hiệu này thường được đưa dưới dạng vector N chiều

• Tập các liên kết: Mỗi liên kết được thể hiện bởi một trọng số (gọi là trọng số liên kết – Synaptic weight). Thông thường các trọng số này được khởi tạo một cách ngẫu nhiên ở thời điểm khởi tạo mạng và được cập nhật liên tục trong quá trình học mạng.

• Bộ tổng (Summing function): thường dùng để tính tổng của các tích đầu vào với trọng số liên kết của nó

• Ngưỡng (còn gọi là một độ lệch – bias): Ngưỡng này thường được đưa vào như một thành phần của hàm truyền

• Hàm truyền (Transfer function): Hàm này được dùng để giới hạn phạm vi đầu ra của mỗi neuron. Nó nhận đầu vào là kết quả của hàm tổng và ngưỡng đã cho

• Đầu ra: (Output) Là tín hiệu đầu ra của mỗi neuron, với mỗi neuron sẽ có tối đa là một đầu ra

**3.HỌC CÓ GIÁM SÁT VÀ HỌC KHÔNG CÓ GIÁM SÁT**

Một mạng neuron cần được đào tạo trước khi nó có thể được đưa vào sử dụng. Huấn luyện liên quan đến việc đưa vào mạng neuron mẫu huấn luyện và cho phép nó tìm hiểu bằng cách hiệu chỉnh trọng số liên kết và các thông số khác nhau. mạng nơ-ron có thể được phân loại thành hai loại dựa vào loại học tập.

## **3.1 MẠNG NEURON HỌC CÓ GIÁM SÁT**

Trong phương pháp học có giám sát, mạng neuron học từ các mẫu. Tập huấn luyện bao gồm một tập hợp các mẫu đầu vào và các kết quả đầu ra mong muốn tương ứng với mẫu đầu vào. Các mạng neuron điều chỉnh trọng số liên kết của nó để tìm hiểu mối quan hệ giữa các cặp đầu vào-đầu ra. Mạng neuron được huấn luyện thành công thì có thể hể được sử dụng để tìm đầu ra phù hợp nhất đối với bất kỳ đầu vào hợp lệ.

Mục tiêu của việc học có giám sát một mô hình toàn cục là tìm ra một hàm f, khi cho sẵn một tập các điểm có dạng (x, f(x)).

**3.2 MẠNG NEURON HỌC KHÔNG CÓ GIÁM SÁT**

Mạng neuron chỉ nhận được một tập hợp các đầu vào từ môi trường bên ngoài. Nó có vẻ bí ẩn để tưởng tượng những gì các mạng có thể có thể học hỏi từ thiết lập một chỉ số đầu vào. Tuy nhiên, có thể để chính thức chứng minh rằng một mạng lưới không có giám sát có thể xây dựng đại diện của các đầu vào có thể được sử dụng cho việc ra quyết định.

**3.3 MẠNG NEURON HỌC TĂNG CƯỜNG**

Trong [học tăng cường](https://vi.wikipedia.org/wiki/H%E1%BB%8Dc_t%C4%83ng_c%C6%B0%E1%BB%9Dng), dữ liệu x hường không được cho 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 và một chi phí tức thời ct, theo một quy trình động nào đó (thường là không được biết). Mục tiêu là tìm 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 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 là một phần của thuật toán toàn cục. Các bài toán thường được giải quyết bằng học tăng cường là các bài toán [điều khiển](https://vi.wikipedia.org/wiki/%C4%90i%E1%BB%81u_khi%E1%BB%83n), [trò chơi](https://vi.wikipedia.org/wiki/Tr%C3%B2_ch%C6%A1i), và các nhiệm vụ [quyết định tuần tự](https://vi.wikipedia.org/w/index.php?title=Quy%E1%BA%BFt_%C4%91%E1%BB%8Bnh_tu%E1%BA%A7n_t%E1%BB%B1&action=edit&redlink=1) (sequential decision making) khác.

**4. GIẢI THUẬT LAN TRUYỀN NGƯỢC**

Để huấn luyện mạng neuron ta cung cấp cho nó một bộ huấn luyện và cho phép nó học bằng cách điều chỉnh trọng số của các liên kết mạng. Một tập huấn luyện là một tập hợp mẫu huấn luyện.

**Training Set = Set of training samples**

Một mẫu đào tạo là một cặp của một vector đầu vào và một mẫu vectơ đầu ra mong muốn. Trong trường hợp đào tạo không có giám sát, các vector đầu ra nên được để null. Chiều dài của vector đầu vào nên được tương tự như số lượng các neuron trong lớp đầu vào, và độ dài vector đầu ra nên được bằng số neuron trong lớp đầu ra.

**Training Sample = (input vector, desired vector)**

Backpropagation thuật toán là một thuật toán giám sát thường được sử dụng để huấn luyện các mạng feed-forward. Nó được giới thiệu lần đầu tiên bởi Paul Werbos trong cuốn sách 'The Roots của Backpropagation'. Ý tưởng cơ bản là xác định mạng neuron hoạt động như thế nào với đầu vào mẫu, so sánh khác nhau như thế nào giữa các hành vi mong muốn và sau đó điều chỉnh trọng số của các liên kết để giảm thiểu sự khác biệt. Quá trình này lặp đi lặp lại cho tất cả các mẫu đào tạo trong nhiều lần thiết lập để đảm bảo huấn

luyện phù hợp.

Việc huấn luyện mạng MLP bởi thuật toán lan truyền ngược sai số bao gồm hai quá trình: Quá trình truyền thẳng và quá trình truyền ngược. Trong quá trình truyền thẳng, các vector đầu vào sẽ được cung cấp cho các neuron của mạng và tín hiệu sẽ được lan truyền lần lượt trên từng lớp mạng. Cuối cùng ta sẽ tính được một tập các đầu ra thực sự của mạng.

Trong suốt quá trình truyền thẳng, tất cả các trọng số liên kết của mạng đều cố định. Ngược lại, trong quá trình truyền ngược, tất cả các trọng số liên kết đó sẽ được hiệu chỉnh theo các luật hiệu chỉnh trọng số. Sai số của mạng sẽ được đo bằng độ sai lệch giữa đầu ra thu được với các giá trị mục tiêu tương ứng. Các sai số này sau đó sẽ được lan truyền ngược lần lượt trên các lớp mạng (từ lớp cuối cùng đến lớp đầu tiên). Các trọng số liên kết sẽ được hiệu chỉnh sao cho các đầu ra thực sự của mạng càng gần với các giá trị mục tiêu càng tốt.

# **5. LEARNING RATE**

Learning rate là một trong những thông số mà điều chỉnh việc làm thế nào để một mạng noron học nhanh và làm thế nào để việc huấn luyện hiệu quả.

Hãy xem xét một neuron mà đang trải qua quá trình học tập. Giả định rằng trọng số của một số liên kết trong mạng một phần được đào tạo là 0,3. Khi mạng được giới thiệu một mẫu huấn luyện mới, thuật toán huấn luyện yêu cầu các liên kết thay đổi trọng số của nó đến 0,7 để nó có thể học các mẫu mới phù hợp. Nếu chúng ta cập nhật trọng số ngay lập tức, các mạng neuron chắc chắn sẽ học các mẫu mới, nhưng nó có xu hướng quên đi tất cả

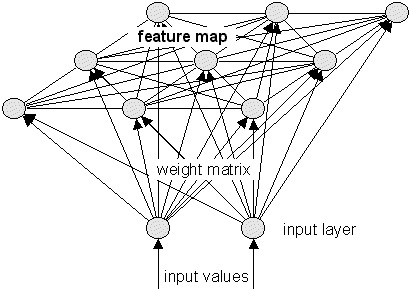
các mẫu nó đã học trước đó. Điều này là do trọng số hiện tại (0,3) là kết quả của tất cả việc học mà nó đã trải qua cho đến nay.

Vì vậy, chúng ta không trực tiếp thay đổi trọng số tới 0,7. Thay vào đó, chúng ta tăng nó bởi một phần nhỏ (chọn 25%) của sự thay đổi cần thiết. Vì vậy, trọng số liên kết của được thay đổi thành 0,4 và chúng ta chuyển sang mẫu đào tạo tiếp theo. Yếu tố này (0,25 trong trường hợp này) được gọi là Learning Rate. Căn cứ theo cách này, tất cả các mẫu huấn luyên được huấn luyện trong một số thứ tự ngẫu nhiên. Khi chu trình đào tạo lặp đi lặp lại nhiều lần, cuối cùng mạng neuron học tất cả các mẫu có hiệu quả.

Learning rate là một giá trị trong khoảng từ 0 đến 1. Chọn một giá trị rất gần bằng không, đòi hỏi một số lượng lớn các chu trình huấn luyện. Điều này làm cho quá trình huấn luyện rất chậm. Mặt khác, nếu learning rate rất lớn, trọng số khác nhau và độ lệch hàm mục tiêu dao động lớn và mạng đạt đến một trạng thái mà việc huấn luyện diễn ra vô ích.

**6. MỘT SỐ KIỂU MẠNG NEURON NHÂN TẠO**

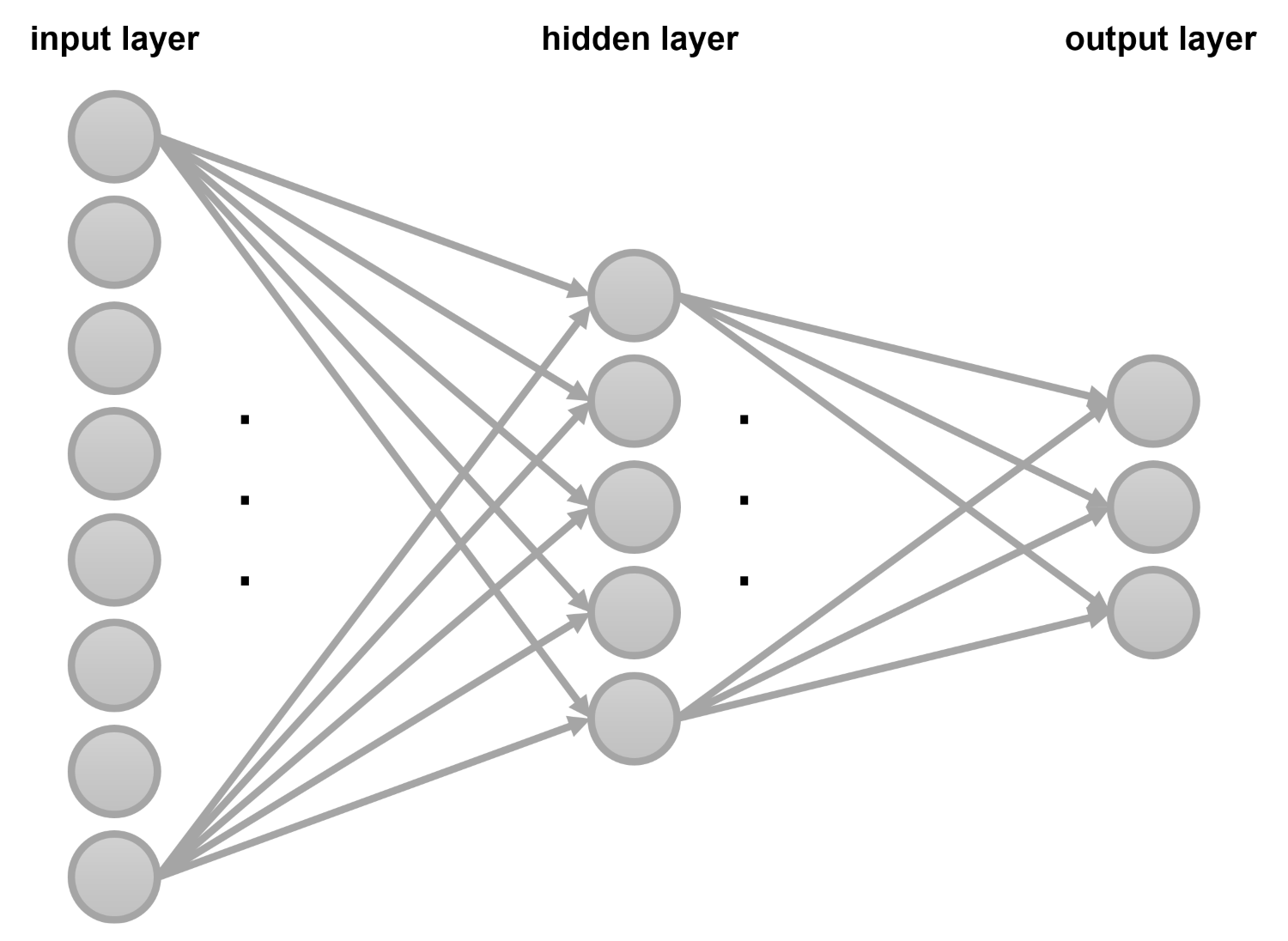
## **6.1 Mạng tự tổ chức (Kohonen feature maps)**

**“Kohonen feature maps”** là mạng nơ ron truyền thẳng sử dụng giải thuật học không giám sát, thông qua một quá trình tự tổ chức, cấu hình các đơn vị đầu ra thành một sơ đồ hình học hoặc không gian. Các mạng Kohonen thể hiện dưới dạng của “sơ đồ tự tổ chức” (SOM - Self-organized maps), bản thân nó thể hiện một lớp đặc biệt của các mạng nơ ron. Mục đích của SOM là biến đổi một tín hiệu đầu vào có số chiều cao, phức tạp thành một sơ đồ rời rạc có số chiều thấp đơn giản hơn. Do vậy, các SOM thường thích hợp cho việc phân tích theo cụm, ở đây, các mẫu ẩn phía dưới trong các bản ghi và các trường được tìm kiếm. Các mạng SOM cấu trúc các nút đầu ra thành các cụm các nút, ở đây các nút trong sự xấp xỉ gần sát hơn tương đương với ****nhau hơn là các nút khác ở xa hơn.

*Kiến trúc thông thường của SOM*.

## **6.2 Mạng nơ ron truyền thẳng đa tầng (Multilayer Perceptrons).**

Mạng truyền thẳng đa tầng là một trong các lớp phổ biến và quan trọng nhất trong các ứng dụng của mạng nơ ron. Mạng neuron này thông thường bao gồm một tập các đầu vào mà hình thành nên tầng đầu vào của mạng, một hoặc nhiều hơn các tầng ẩn của các nút tính toán, và cuối cùng một tầng đầu ra của các nút tính toán. Việc xử lý là theo một hướng thẳng dựa trên cơ sở từng lớp. MLP đã được ứng dụng thành công để giải quyết một số các bài toán khó và đa dạng bằng việc huấn luyện mạng dưới dạng có giám sát với một thuật giải được nhiều người biết đến là giải thuật lỗi lan truyền ngược (Back-propagation).



*Kiến trúc thông thường của Multilayer Perceptrons*

# **7. THIẾT KẾ MẠNG NEURON**

Thiết kế một mạng neuron nhân tạo cho một ứng dụng cụ thể liên quan đến việc lựa chọn đúng loại mạng, tìm một số thích hợp của các lớp ẩn, phương pháp thích hợp để khởi tạo trọngsố, thuật toán học thích hợp, các thế đào tạo, tỷ lệ học tập và số lượng mẫu đào tạo để sử dụng. Hầu hết các thông số này đều phụ thuộc vào ứng dụng mà các mạng neuron đang được thiết kế.

Dưới đây là một số hướng dẫn chung về thiết kế mạng nơrron:

**7.1Số lượng lớp ẩn (trong mạng Backpropagation):**

Một mạng lưới Backpropagation không có lớp ẩn không thể thực hiện phân loại không tuyến tính. Vì vậy, một trong những lớp ẩn là phải cho một mạng lưới backpropagation.

Hơn nữa, nó đã được toán học đã chứng minh rằng một mạng lưới backpropagation với lớp ẩn duy nhất khi đào tạo phù hợp, có thể dùng để xấp xỉ hàm. Vì vậy, lớp ẩn duy nhất là sự lựachọn tốt nhất trong hầu hết trường hợp.

Có nhiều lớp ẩn tăng tốc quá trình học tập và mạng được đào tạo phù hợp chính xác với các mẫu đào tạo nhưng không thực hiện tốt trên các dữ liệu thử nghiệm. Hiệu ứng này được gọi là overtraining nơi mạng lưới huấn luyện có xu hướng ghi nhớ các mẫu huấn luyện thay vì học tập chúng.

**7.2 Mạng neuron ban đầu**

Khởi tạo đúng trọng số liên kết neuron có ảnh hưởng lớn tới việc đào tạo tốc độ cũng như xác định hiệu quả của đào tạo. Thông thường, trọng số được khởi tạo với giá trị ngẫu nhiên từ -0,5 đến +0,5 (giá trị cao có xu hướng kết quả trong khu vực bão hòa sau khi kích hoạt, các giá trị ban đầu nhỏ thì ra các giá trị gần bằng không).

Mạng neuron thực hiện quá trình khởi tạo như là một module pluggable. Tuỳ chỉnh các thuật toán khởi tạo có thể được cắm vào bằng cách thực hiện Initializer giao diện.

**7.3 Số mẫu huấn luyện**

Quyết định như thế nào để các mẫu đại diện cho các chức năng huấn luyện thực tế là tốt nhất. Thông thường, các lỗi học hơi tăng so với sự gia tăng kích thước của tập huấn luyện, đồng thời ta cũng có thể nhận thấy rằng giảm lỗi và mạng thực hiện tốt hơn về dữ liệu thử nghiệm.

Một mối quan hệ giữa kích thước mạng và số lượng tối ưu của mẫu đào tạo có thể được tìm thấy.

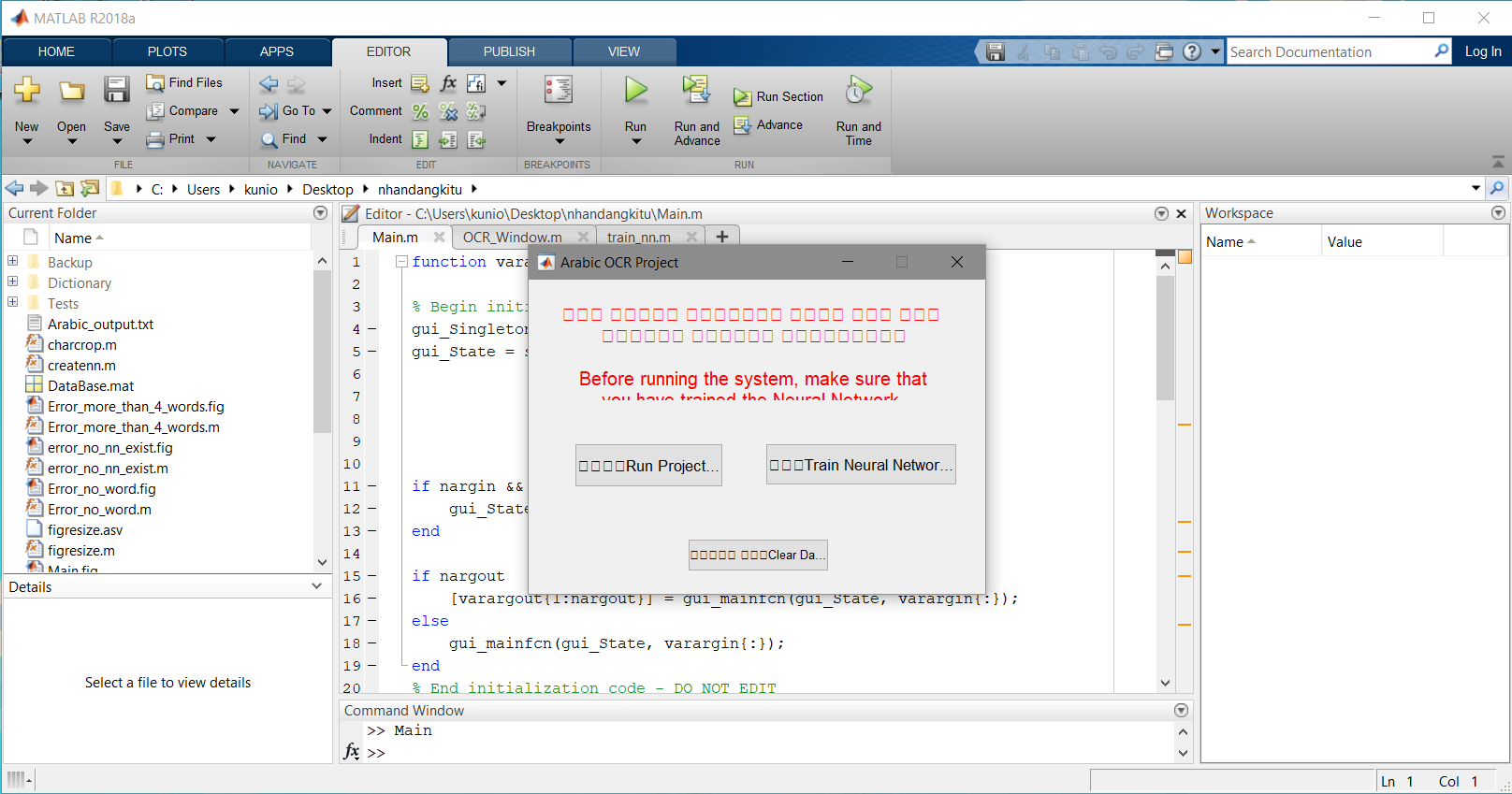
**8. CÀI ĐẶT MÔI TRƯỜNG PHÁT TRIỂN**

* Dự án được viết trên ngôn ngữ C# C++. Do vậy bạn cần phải sử dụng Microsoft .NET framework.
* Sử dụng công cụ phát triển là Visual Studio Express editions (http://www.microsoft.com/Express/ )

**9. Ứng dụng minh họa**

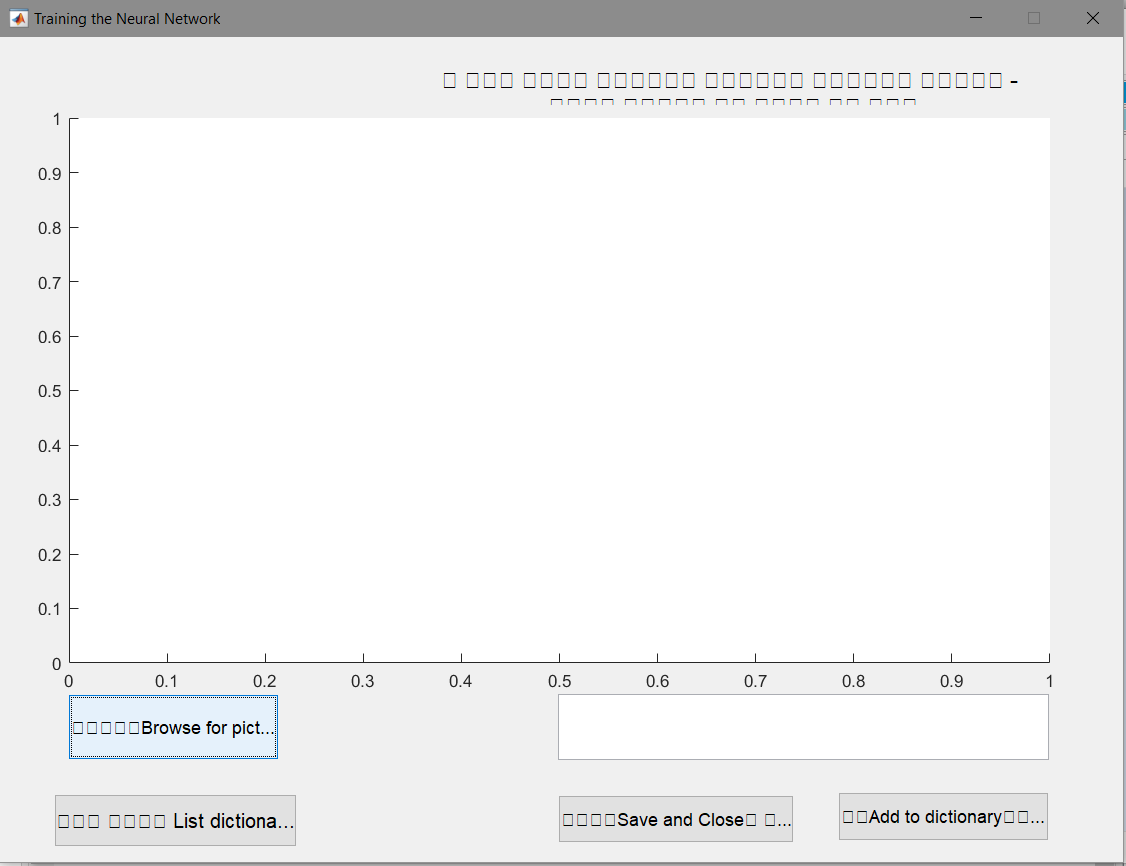
**9.1 Nhận diện kí tự với Neuron bằng Matlab**

Khi khởi chạy ứng dụng thì Matlab sẽ hiển thị hộp thoại như hình dưới và việc đầu tiên là chúng ta cần phải train Neural Network để mới có thể chạy được nhận diện kí tự.

****

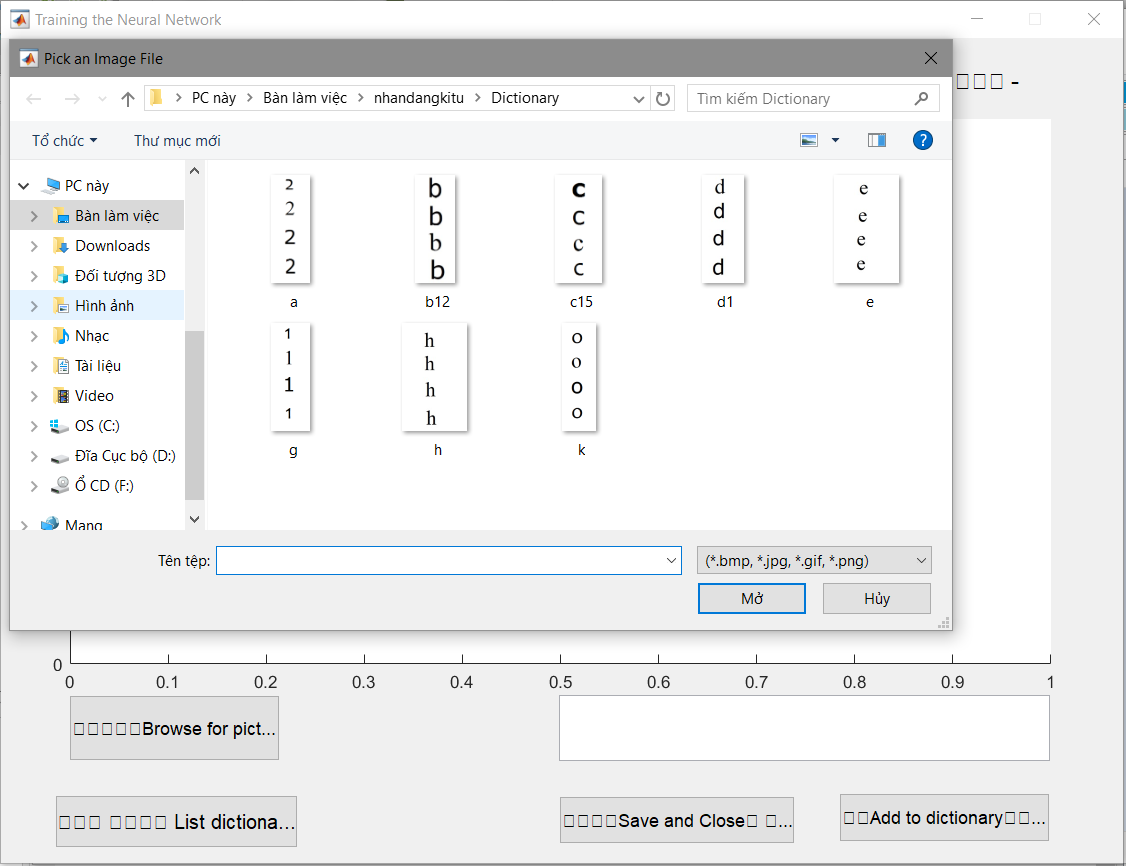
**9.1.1 Giao diện phần mềm nhận dạng kí tự**

Sau khi vào Menu train thì việc đầu tiên cần làm là tìm hình mẫu kí tự ở trong mục “Browser for picture”.

****

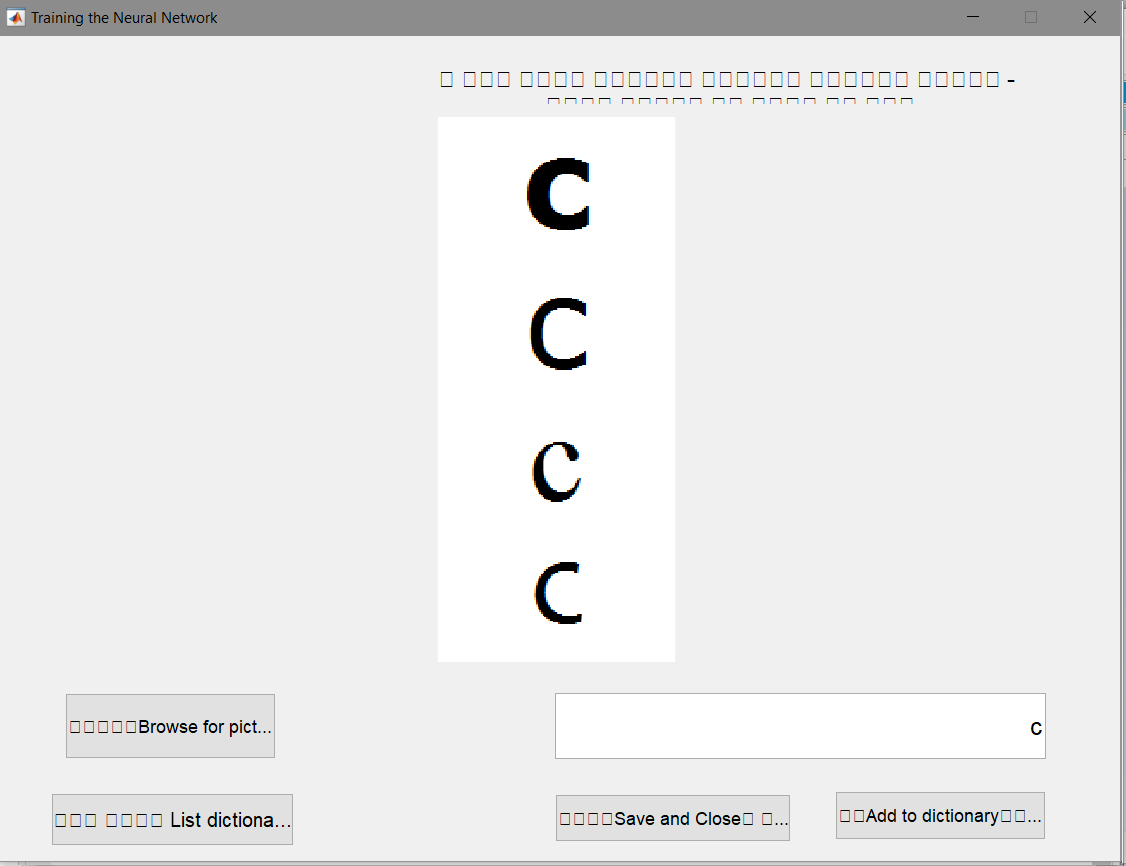
**9.1.2 Giao diện phần mềm nhận dạng kí tự**

Tiếp theo là chọn ngẫu nhiên 1 hình ảnh mẫu đã được thêm sẵn trong Dictionary, và chọn “Open” .



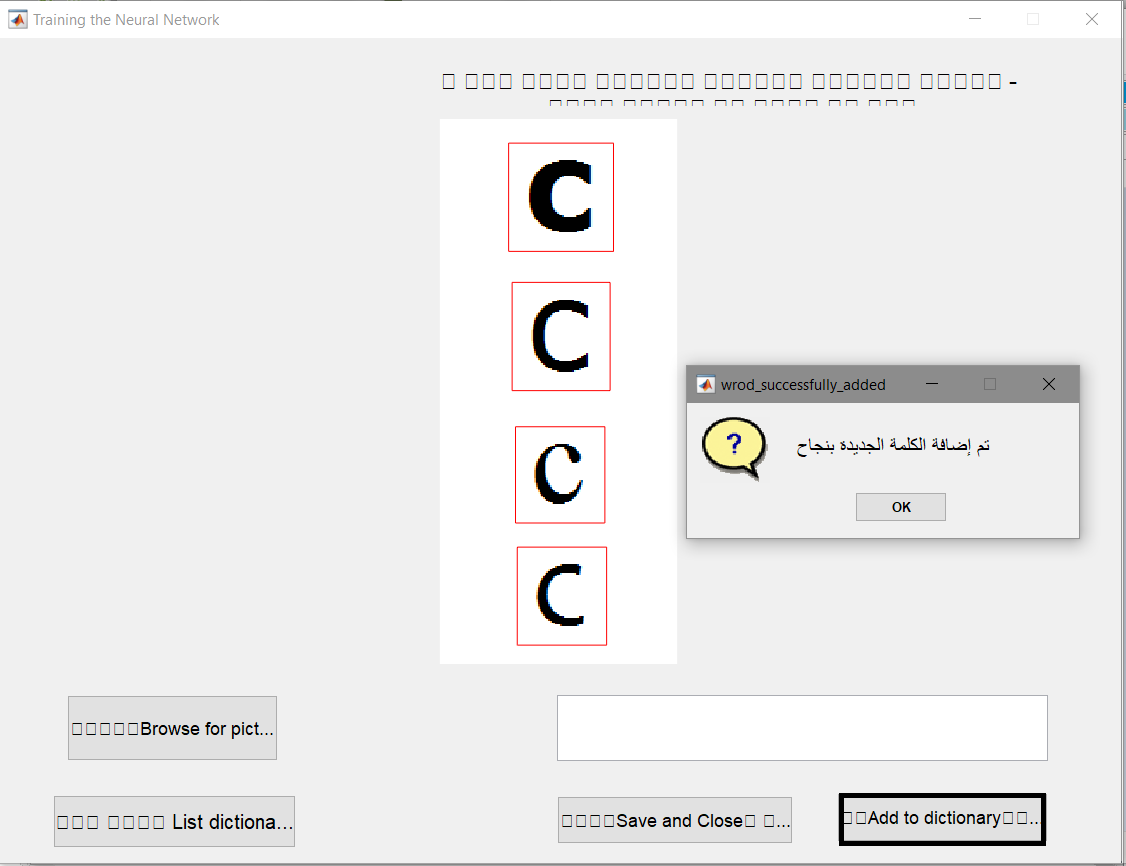
**9.1.3 Giao diện phần mềm nhận dạng kí tự**

Hình mẫu đã được chèn vào thành công và tiếp theo là gõ kí tự như hình mẫu để phần mềm có thể xác định kí tự dựa trên cả 2 hình ảnh và kí tự nhập.



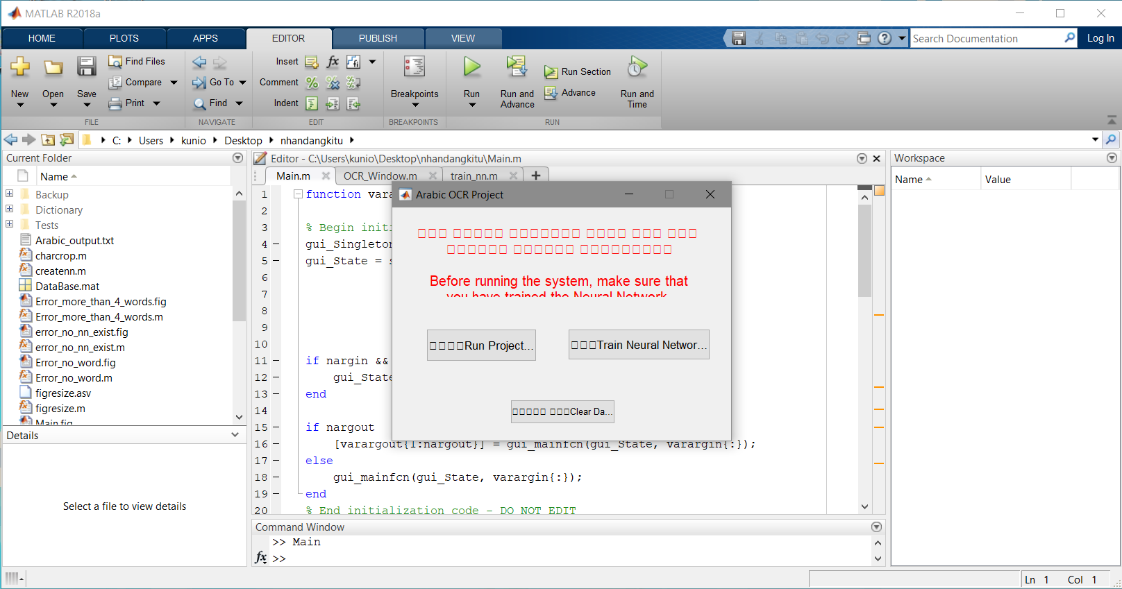
**9.1.4 Giao diện phần mềm nhận dạng kí tự**

Sau khi nhập kí tự xong và click vào “Add to dictionary”, sau đó hộp thoại thành công xuất hiện và nhấn “OK”. Tiến hành Save and close lại và khởi chạy phần mềm chính.



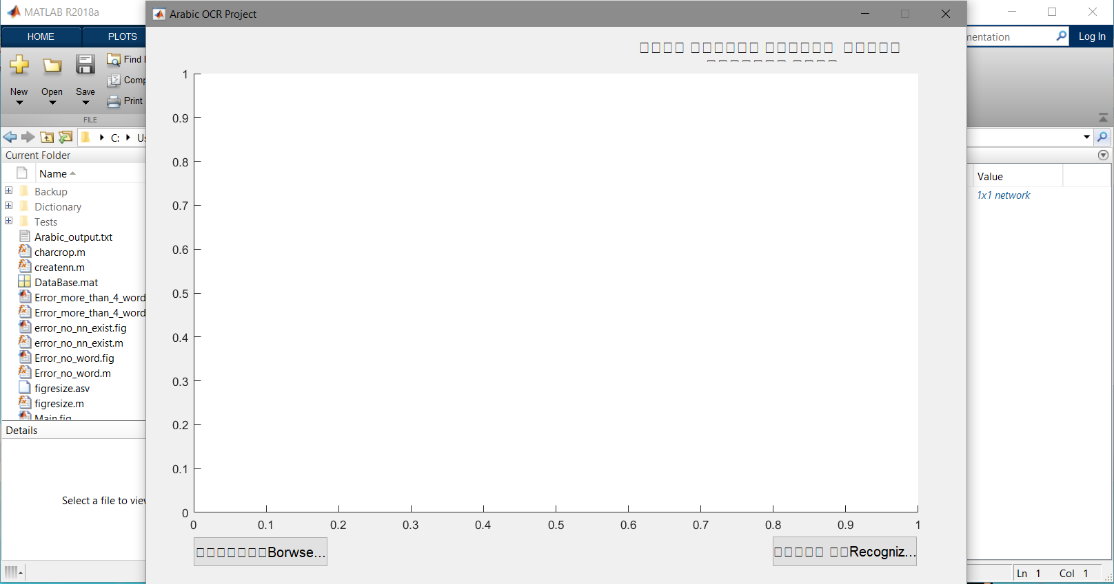
**9.1.5 Giao diện phần mềm nhận dạng kí tự**

Trở lại với giao diện đã được nêu ở hình 9.1.1 và lần này click vào “Run Project” để tiến hành nhận diện chữ

****

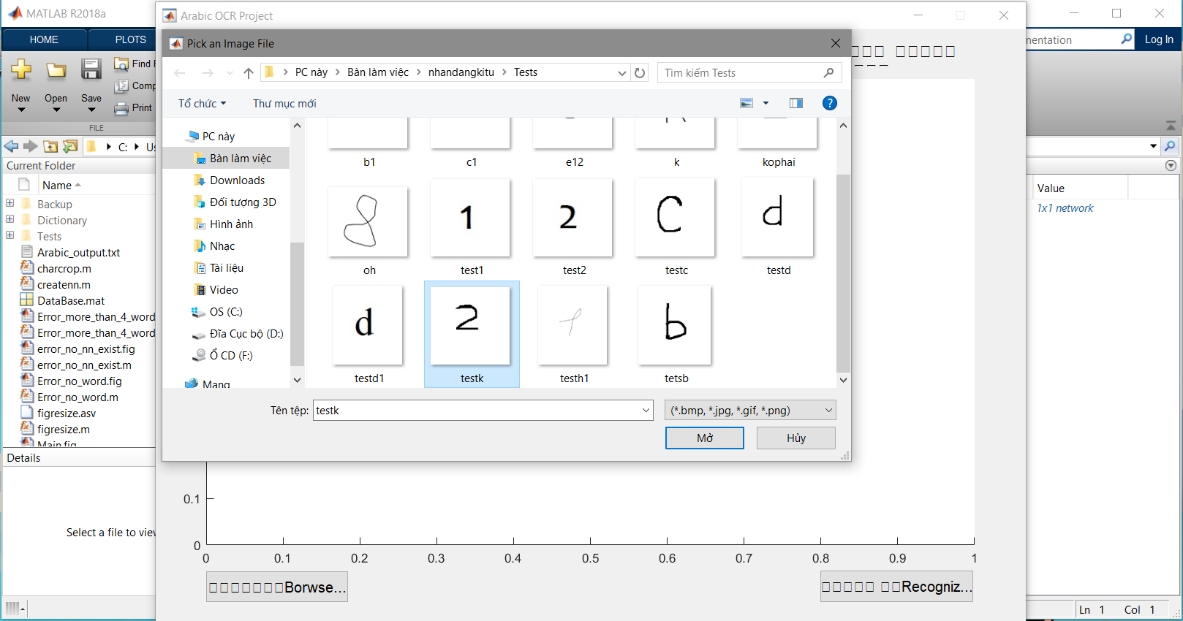
**9.1.5 Giao diện phần mềm nhận dạng kí tự**

Giao diện chính xuất hiện và trước tiên là phải nhập hình để phần mềm tự nhận diện kí tự bằng cách nhấn vào “Browser for picture”

****

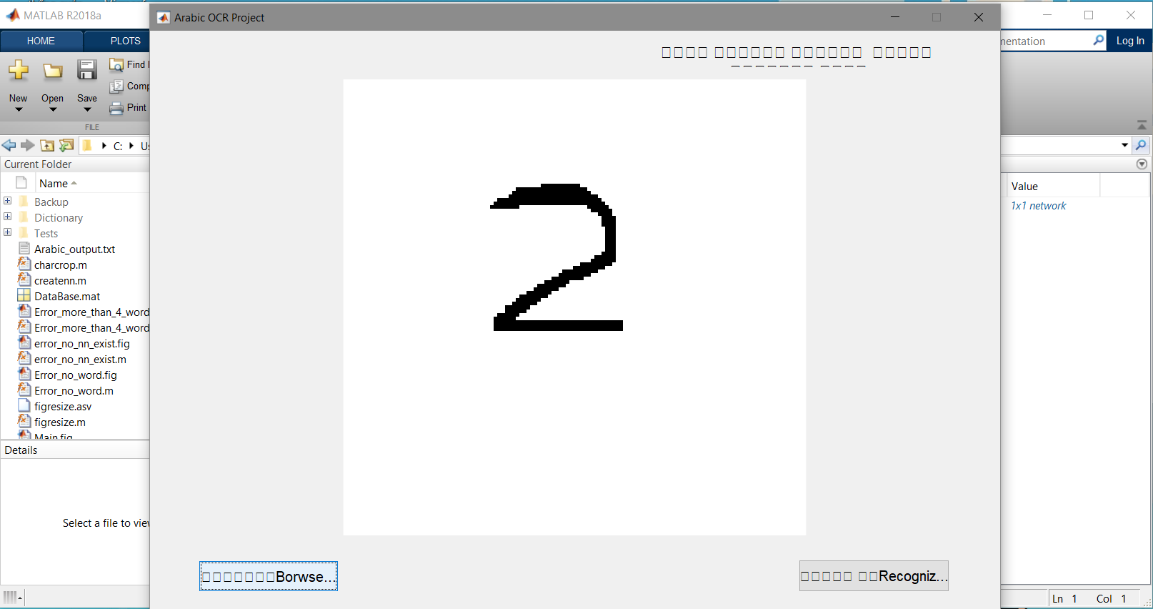
**9.1.5 Giao diện phần mềm nhận dạng kí tự**

Giao diện chọn hình ảnh xuất hiện và chọn ngẫu nhiên bất kì kí tự nào để phần mềm nhận diện.



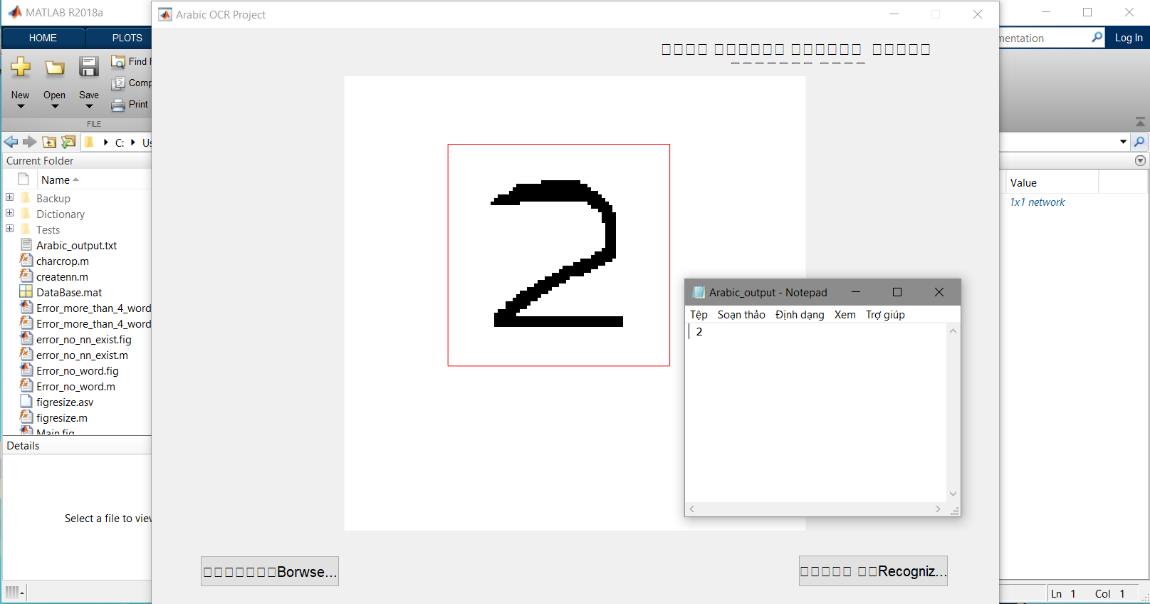
**9.1.5 Giao diện phần mềm nhận dạng kí tự**

Sau khi nạp hình ảnh thành công thì tiến hành đến bước cuối cùng để nhận kết quả phần mềm nhận diện trả về bằng cách click vào “Recognize”



**9.1.5 Giao diện phần mềm nhận dạng kí tự**

Sau khi click vào “Recognize” thì kí tự đã được phần mềm nhận diện kí tự trả về kết quả vào trong Notepad.

****

**9.1.5 Giao diện phần mềm nhận dạng kí tự**