**­­­­­­­­­BỘ GIÁO GIỤC VÀ ĐÀO TẠO**

**TRƯỜNG ĐẠI HỌC NHA TRANG**

**KHOA CÔNG NGHỆ THÔNG TIN**

****

# ­­

# ĐỒ ÁN TỐT NGHIỆP

**XÂY DỰNG ỨNG DỤNG HỌC TIẾNG ANH DÙNG GIẢI THUẬT PHÂN LỚP DỮ LIỆU**

**Giảng viên hướng dẫn: Phạm Thị Kim Ngoan**

**Sinh viên thực hiện: NGUYỄN TẤT CHỦ**

**55133917 – 55.CNTT1**

*Nha Trang –2017*

LỜI CẢM ƠN

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.

…

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.

MỤC LỤC

DANH MỤC CÁC KÝ HIỆU, CÁC CHỮ VIẾT TẮT

|  |  |  |  |
| --- | --- | --- | --- |
| **STT** | **Ký hiệu viết tắt** | **Từ gốc** | **Diễn giải** |
| 1 | I/O | Input/Ouput |  |
| 2 | API | Application Programming Interface |  |
| 3 | JSON | JavaScript Object Noattion |  |
| 4 | IDE | Integrated Development Environment |  |
| 5 | MVC | Model – View – Controller |  |
| 6 | MVVM | Model – View – View Model |  |
| 7 | MVW | Model – View – Whatever | Mô hình kết hợp |

DANH MỤC CÁC BẢNG

DANH MỤC CÁC HÌNH

# CHƯƠNG 1: TỔNG QUAN

## 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, đề tài này chỉ xin trình bày những vấn đề có liên quan, những kiến thức được sử dụng phục vụ cho việc hoàn thành đồ án*

* *Bài toán làm gì*
* *Phương pháp nghiên cứu*
* *.*

# CHƯƠNG 2: CƠ SỞ LÝ THUYẾT

## Nền tảng NodeJS

Node.js là một nền tảng dựa vào Chrome Javascript runtime để xây dựng các ứng dụng nhanh, có độ lớn. Node.js sử dụng các phần phát sinh các sự kiện (event-driven), mô hình non-blocking I/O để tạo ra các ứng dụng nhẹ và hiệu quả cho các ứng dụng về dữ liệu thời gian thực chạy trên các thiết bị phân tán.

NodeJS là một mã nguồn mở, đa nền tảng được thiết kế để xây dựng các ứng dụng Internet nhanh, có độ lớn, có khả năng mở rộng phía máy chủ dựa trên Chrome Javascript Engine V8 được phát triển bởi Ryan Dahl năm 2009 dưới sự bảo trợ của Joyent.

Những môi trường tương tự được viết trong các ngôn ngữ khác bao gồm Twisted cho Python, Perl Object Environment cho Perl, libevent cho C và EventMachine cho Ruby. Khác với hầu hết các chương trình Javascript, Nodejs không chạy trên một trình duyệt mà chạy trên Server. Node.js sử dụng nhiều chi tiết kỹ thuật của CommonJS. Nó cung cấp một môi trường REPL cho kiểm thử tương tác.

NodeJS cung cấp cho nhà phát triển các module JavaScript đa dạng, giúp cho việc phát triển ứng dụng dễ dàng hơn, tuy nhiên NodeJS chỉ là môi trường, điều này có nghĩa là nhà phát triển phải làm mọi thứ. Không có một máy chủ nào mặc định, một đoạn script sẽ xử lý tất cả các kết nối từ máy khách. Điều này làm giảm được đáng kể tài nguyên được sử dụng trong ứng dụng.

### Đặc điểm của NodeJS

Một vài đặc điểm quan trọng của NodeJS khuyến cho NodeJS trở thành lựa chọn hàng đầu cho các nhà phát triển phần mềm:

**Không đồng bộ và Phát sinh sự kiện (Event Driven):** Khác với ngôn ngữ C#, PHP hay Java, tất các các APIs của thư viện Node.js đều bất đồng bộ (non-blocking). Điều này rất cần thiết vì Node.js không bao giờ đợi một API trả về dự liệu. Server chuyển sang một API sau khi gọi nó và có cơ chế thông báo về Sự kiện của Node.js giúp Server nhận đựa phản hồi từ các API gọi trước đó.

**Chạy rất nhanh:** Dựa trên V8 Javascript Engine của Google Chrome, thư viện Node.js rất nhanh trong các quá trình thực thi các đoạn code.

**Các tiến trình đơn giản nhưng hiệu năng cao:** Node.js sử dụng một mô hình luồng đơn (single thread) với các sự kiện lặp. Các cơ chế sự kiện giúp Server trả lại các phản hồi với một cách không khóa và tạo cho Server hiệu quả cao ngược lại với các cách truyền thống tạo ra một số lượng luồng hữu hạn để quản lý request. Nodejs sử dụng các chương trình đơn luồng và các chương trình này cung cấp các dịch vụ cho số lượng request nhiều hơn so với các Server truyền thống như Apache HTTP Server.

**Không đệm:** Ứng dụng NodeJS không lưu trữ các dữ liệu buffer.

**Có giấy phép:** mã nguồn mở NodeJS (https://github.com/nodejs/node) được phát hành dựa vào MIT License.

Với những đặc điểm này NodeJS là một sự lựa chọn hoàn hảo cho các lĩnh vực gồm:

* Các ứng dụng về I/O
* Các ứng dựng về luồng dữ liệu
* Các ứng dụng về dữ liệu hướng đến thời gian thực
* Các ứng dụng dựa vào JSON APIs
* Các ứng dụng Single Page Application

Hiện tại có rất nhiều hệ thống lớn hiện tại đang sử dụng NodeJS, trong số này có thể kể đến như: eBay, GE, GoDaddy, Microsoft, Yahoo, Linkedin, Paypal,Uber,…

### Cài đặt môi trường NodeJS và các công cụ phát triển

Để cài đặt môi trường phát triển NodeJS, có hai cách là tải phiên bản cài đặt từ trang chủ <https://nodejs.org/en/download/> theo hệ điều hành tương ứng hoặc sử dụng các công cụ hỗ trợ cài đặt gói thông qua dòng lệnh:

* Trên Windows sử dụng Chocolatey(<https://chocolatey.org)>: *choco install nodejs*
* Trên MacOS sử dụng Homebrew(<https://brew.sh)>: *brew install node*
* Trên Linux sử dụng trình cài đặt gói mặc định: *sudo apt-get install nodejs*

Sau khi cài đặt sử dụng lệnh *node –*v, để kiểm tra đã cài đặt thành công hay chưa, nếu thành công dòng lệnh này sẽ in ra màn hình console phiên bản NodeJS đã cài trên hệ thống.

Ngoài điều kiện tiên quyết là cài đặt môi trường, ngoài các ứng dụng soạn thảo mặc định của hệ đièu hành để thuận tiện cho quá trình phát triển cần có thêm những trình soạn thảo code (Editor) hoặc IDE, các IDE miễn phí và nổi bật thường được các lập trình viên trên thế giới sử dụng bao gồm: Visual Studio Code, Sublime Text, Notepad++, …

### Phát triển ứng dụng với NodeJS

Bất kỳ lập trình viên nào muốn xây dựng ứng dụng với NodeJS cũng cần phải biết và hiểu được các khái niệm như module, lập trình không đồng độ, function callback, promise,..; cũng như các đối tượng, các lớp, các hàm được cung cấp sẵn của framework này. Phần dưới đây sẽ đưa ra những định nghĩa và ví dụ chi tiết.

**Modules** trong NodeJS giống như các thư viện trong C, C#, Java, … mỗi module là một đối tượng (Object) chứa các hàm, các hằng số, … Điều này làm cho việc xây dựng các ứng dụng phức tạp trở nên đơn giản hơn. Mỗi module đóng gói tập các phương thức, thông tin, … liên quan đến một đối tượng, chức năng của Module. Ví dụ **fs** là Module chứa các hàm, các hằng số cụ thể liên quan đến việc đọc ghi dữ liệu hệ thống. Ngoài những module được NodeJs cung cấp sẵn để thao tác với hệ thống và một số tiện ích khác. NodeJs cho phép bất kỳ ai cũng có thể tạo ra module của riêng mình, hơn thế nữa NodeJs đi kèm với trình quản lý gói **npm** (<https://www.npmjs.com)> giúp cho việc cài đặt tích hợp các module vào ứng dụng của mình trở nên vô cùng đơn giản.

Các thao tác cần thiết để sử dụng một module đó là cài đặt module và tham chiếu tới module.

* Cú pháp cài đặt bằng npm: **npm install** *<tên package>.*Ví dụ: *npm install http*
* Cú pháp gọi tham chiếu module: **var** <*tên biến> =* **require(‘***<tên module>***’)**.   
  Ví dụ: *var http = requỉe(‘http’)*

Trong ví dụ trên hàm *require* trả về tham chiếu tới module *http* được cài đặt từ internet về thông qua lệnh *npm install*. Để tham chiếu tới một Module cục bộ do trên hệ thống phải chỉ rõ đường dẫn tới tập tin của Module đó. Ví dụ: *var myModule = requỉe(‘./moduleName.js’)*

Có thê xem mỗi module là một đoạn mã được đóng gói lại với nhau, mã lệnh bên trong module có phạm vi là *private*. Gần giống như việc public một hàm của một Class, bằng việc sử dụng đối tượng exports, sẽ giúp các hàm, các hằng private có thể để đưa ra ngoài sử dụng cho trên ứng dụng. Xem xét ví dụ dưới đây:

const PI = **Math**.PI;

**exports**.dienTich = function (r) {

return PI \* r \* r;

};

**exports**.chuVi = function (r) {

return 2 \* PI \* r;

};

Đoãn mã trên tạo ra hằng số *PI* và hai function, hằng số *PI* chỉ có thể sử dụng trong nội bộ module đang định nghĩa ở trên. Qua việc sử dụng đối tượng ***exports***, hai hàm *dienTich* và *chuVi* có thể sử dụng khi tham chiếu tới Module này.

**Global Object**: các object có sẵn ở tất cả các module bao gồm: Buffer, \_\_dirname, \_\_filename, console, exports, global, module, process; các function có sẵn bao gồm: require, setTimeout, … và một class có sẵn là Buffer.

**Asynchronous** (lập trình không đồng bộ): Không giống như các ngôn ngữ lập trình truyền thống C#, Java, C++, chương trình sẽ chạy tuần tự từng lệnh và chỉ thực hiện lệnh tiếp theo khi lệnh trước đó đã thực hiện xong, điều này sẽ sinh ra một tráng thái hay gọi là trạng thái chờ, Javascript là ngôn ngữ lập trình bất đồng bộ, khi thực thi chương trình có thể bỏ qua một số bước chờ không cần thiết, thực hiện nhiều công việc song song cùng lúc.

Lập trình bất đồng bộ có

Modules

Debugger

Console

Cluster

Add-ons

Buffer

Error Handling

Callbacks

Streaming

DNS

Domain

Global

Net

Crypto

## Angular Framework

AngularJS là một framework viết bằng Javascript được sử dụng phía client có cấu trúc cho các ứng dụng web động. Nó cho phép bạn sử sụng HTML như là ngôn ngữ mẫu và cho phép mở rộng cú pháp của HTML để diễn đạt các thành phần ứng dụng một cách rõ ràng súc tích.

AngularJS là framework mã nguồn mở hoạt động dưới giấy phép Apache Lincense được đưa ra lần đầu năm 2009 bởi Misko Hevery và Adam Abrons. Hiện tại thư viện này được duy trì và phát triển bởi hãng Google.

Các tính năng cốt lõi của Framework AngularJS bao gồm:

* Data-binding: Nó tự động đồng bộ hóa dữ liệu giữa thành phần model và view.
* Scope: Là những đối tượng hướng đến model, nó hoạt động như là cầu nối giữa controller và view
* Controller: Đây là những tính năng của AngularJS mà được giới hạn tới một scope cụ thể
* Service: AngularJS hoạt động với một vài dịch vụ (service) có sẵn , ví dụ $http để tạo XMLHttpRequests. Nó là các singleton object mà được khởi tạo duy nhất một lần trong ứng dụng
* Filter: Nó lựa chọn (hay là lọc) các tập con từ tập item trong các mảng và trả về các mảng mới
* Directive: Directive là các marker trong các phần tử DOM (như các phần tử, thuộc tính, css và nhiều hơn thế). Nó có thể dùng để tạo các thẻ HTML riêng phục vụ những mục đích riêng. AngularJS có những directive có sẵn như ngBind, ngModel, ngController, ngApp…
* Template:Là các rendered view với các thông tin từ controller và model. Nó có thể được sử dụng trong các file riêng rẽ (ví dụ như index.html) hoặc nhiều view với một trang sử dụng “partials”
* Routing: Là khái niệm của sự chuyển dịch qua lại các view
* Deep Linking: Cho phép bạn mã hóa trạng thái các ứng dụng trên địa chỉ URL để nó có thể được bookmark. Các ứng dụng có thể được phục hồi lại từ các địa chỉ URL với cùng một trạng thái
* Dependency Injection: AngularJS có sẵn một hệ thống con dependency injection để giúp các lập trình viên tạo ra các ứng dụng dễ phát triển, dễ hiểu và kiểm tra

AngularJS được thiết kế kết hợp giữa mô hình MVC và MVVM hay còn được là MVW. Những ưu điểm nổi bật của AngularJS đó là:

* AngularJS cung cấp khả năng tạo ra các Single Page Application với API service một cách rất rõ ràng và dễ dàng để duy trì, nâng cấp.
* AngularJS cung cấp khả năng Data binding tới HTML do đó giúp người dùng cảm giác linh hoạt, thân thiện, trực quan.
* AngularJS code dễ dàng trong giai đoạn unit test, functional testing.
* AngularJS sử dụng dependency injection.
* AngularJS cung cấp khả năng tái sử dụng các component (thành phần).
* Với AngularJS, lập trình viên sẽ viết ít code hơn, với nhiều chức năng hơn.
* Với AngularJS, view là thành phần trong trang HTML thuần, trong khi controller được viết bởi JavaScript với quá trình xử lý nghiệp vụ.
* AngularJS có thể sử dụng kết hợp với các framework, thư viện khác mà không gây xung đột

Bên cạnh những ưu điểm, AngularJS cũng có những nhược điểm riêng:

* Không an toàn là một JavaScript Framework, ứng dụng được viết bởi AngularJS nên không an toàn. Vấn đề này được giải quyết nếu phía máy chủ web có cơ chế chứng thực và phân quyền phù hợp
* Được xây dựng bằng JavaScript nên khi người sử dụng vô hiệu hoá tính năng Javascript thì ứng dụng bị vô hiệu hoá.

Các thành phần của AngularJS quan trọng khi xây dựng ứng dụng bao gồm các directive: ngApp, ngController, ngModel,… trong chương tiếp theo sẽ cụ thể hoá các thành phần này trên ứng dụng mà đồ án xây dựng.

## NoSQL và hai hệ quản trị CSDL MongoDB, Redis

### NoSQL

Với các công nghệ phát triển website, hệ quản trị cơ sở dữ liệu quan hệ dựa trên SQL đã thống trị hầu hết các hệ Quản trị Cơ sở dữ liệu. Tuy nhiên thời gian gần đây, một cách tiếp cận mới đã bắt đầu biết đến là NoSQL, tạo ra sự thay thế cho các hệ quản trị cơ sở dữ liệu quan hệ truyền thống

Thuật ngữ NoSQL có nghĩa là Non-Relation hoặc Not Only SQL – không rằng buộc hoặc phi quan hệ, ám chỉ những đến CSDL không dùng mô hình dữ liệu quan hệ để quản lý dữ liệu trong lĩnh vực phần mềm.

Những đặc điểm của CSDL NoSQL :

* Looser consistency: NoSQL tổ chức lưu trữ và truy xuất dữ liệu theo cơ chế “thoáng hơn trong đảm bảo tính nhất quán của dữ liệu” so với mô hình dữ liệu quan hệ tryền thống nhằm cải thiện hiệu suất, đảm bảo dữ liệu luôn được đáp ứng tốt hơn
* Eventual consistency: NoSQL không yêu cầu phải đảm bảo tính nhất quán của dữ dữ liệu ngay tức thì mà sẽ hiện thực tính nhất quán của dữ liệu theo cơ chế lan truyền
* Distributed storage: hay còn gọi là lưu trữ phân tán là thay vì dữ liệu được lưu trữ trên một máy chủ duy nhất thì hệ thống sẽ lưu dữ liệu trên nhiều máy khác nhau
* Horizontal scalable: hay còn gọi là khả năng mở rộng chiều ngang. Bình thường, với các hệ quản trị cơ sở dữ liệu quan hệ , khi mà dữ liệu quá lớn phương pháp tăng khả năng lưu trữ là sẽ phải mở rộng (nâng cấp máy chủ), còn đối với NoSQL thì chỉ cần bổ sung thêm máy chủ khác vì hệ thống hỗ trợ lưu trữ phân tán trên nhiều máy

Ưu điểm của NoSQL:

* NoSQL là mã nguồn mở: điều này có nghĩa là bạn sẽ dễ dàng phát triển một ứng dụng có sử dụng NoSQL mà không phải tốn chi phí license.
* Dễ mở rộng quy mô: NoSQL đã thay thế cách mở rộng quy mô truyền thống của các hệ quản trị cơ sở dữ liệu quan hệ bằng hình thức “mở rộng ra ngoài”. Với hình thức mở rộng đặc biệt này, thay vì phải bổ sung thêm các máy chủ lớn hơn vào hệ thống khi dữ liệu lớn lên, thì NoSQL lại hỗ trợ doanh nghiệp phân tán dữ liệu qua nhiều máy chủ khi dữ liệu gia tăng.
* Hỗ trợ các mô hình dữ liệu khác nhau tuỳ mục đính và cách thức lưu trữ dữ liệu như lưu kiểu key-value, BigTable, lưu document hay lưu thông tin graph

Nhược điểm của CSDL này đó là:

* Thiếu tính tương thích: các CSDL NoSQL khác nhau không tương thích với nhau, mỗi CSDL NoSQL có một giao diện và cung cấp API khác nhau và chưa có một tiêu chuẩn chung nào. Điều này có nghĩa là bạn sẽ gặp khó khăn trong việc chuyển từ nhà cung cấp này sang nhà cung cấp khác
* Hạn chế về nghiệp vụ: NoSQL hiện chưa hỗ trợ các dạng phân tích dữ liệu lớn và mạnh mẽ mà các doanh nghiệp đã quen thuộc trong các RDBMS

Phân loại các CSDL NoSQL

Có nhiều cách phân loại các cơ sở dữ liệu NoSQL khác nhau, mỗi loại với các loại và loại con khác nhau, một số trong số đó có thể chồng chéo lên nhau. Dưới đây là một phân loại cơ bản dựa trên mô hình dữ liệu, chia các hệ quản trị CSDL thành các nhóm khác nhau:

* Column: Accumulo, Cassandra, Druid, HBase, Vertica
* Document: Apache CouchDB, Clusterpoint, Couchbase, DocumentDB, HyperDex, Lotus Notes, MarkLogic, MongoDB, OrientDB, Qizx, RethinkDB
* Key-value: Aerospike, CouchDB, Dynamo, FairCom c-treeACE, FoundationDB, HyperDex, MemcacheDB, MUMPS, Oracle NoSQL Database, OrientDB, Redis, Riak, Berkeley DB
* Graph: AllegroGraph, InfiniteGraph, MarkLogic, Neo4J, OrientDB, Virtuoso, Stardog
* Multi-model: Alchemy Database, ArangoDB, CortexDB, FoundationDB, MarkLogic, OrientDB

Một phân loại chi tiết hơn như sau, dựa trên cách phân loại của Stephen Yen[[1]](#endnote-1):

|  |  |
| --- | --- |
| **Loại** | **Hệ quản trị CSDL** |
| Key-Value Cache | Coherence, eXtreme Scale, GigaSpaces, GemFire, Hazelcast, Infinispan, JBoss Cache, Memcached, Repcached, Terracotta, Velocity |
| Key-Value Store | Flare, Keyspace, RAMCloud, SchemaFree, Hyperdex, Aerospike |
| Key-Value Store (Eventually-Consistent) | DovetailDB, Oracle NoSQL Database, Dynamo, Riak, Dynomite, MotionDb, Voldemort, SubRecord |
| Key-Value Store (Ordered) | Actord, FoundationDB, Lightcloud, LMDB, Luxio, MemcacheDB, NMDB, Scalaris, TokyoTyrant |
| Máy chủ Cấu trúc dữ liệu | Redis |
| Tuple Store | Apache River, Coord, GigaSpaces |
| Cơ sở dữ liệu đối tượng | DB4O, Objectivity/DB, Perst, Shoal, ZopeDB |
| Document Store | Clusterpoint, Couchbase, CouchDB, DocumentDB, Lotus Notes, MarkLogic, MongoDB, Qizx, RethinkDB, XML-databases |
| Wide Column Store | BigTable, Cassandra, Druid, HBase, Hypertable, KAI, KDI, OpenNeptune, Qbase |

Bảng 1: Bảng phân loại hệ quản trị CSQL NoSQL

### Hệ quản trị CSDL MongoDB

### Hệ thống lưu trữ cache máy chủ Redis

## Thuật toán phân lớp dữ liệu Neural Network

## Xử lý ảnh và các phương pháp trích chọn đặc trưng

* NODEJS
* AngularJS, Bootstrap
* MongoDB
* Redis
* THUẬT TOÁN PHÂN LỚP (MẠNG NEURAL, SVM, RANDOM FOREST, DNN)
  + Neural sinh học và ý tưởng xây mạng neural nhân tạo
    - Tham khảo file Thelf/Report/ 2278500-xu-ly-anh-140707223606-phpapp02.pdf **p.68-p.75**
  + Neural nhận tạo và luật học
  + Mô hình và cấu trúc mạng neural nhân tạo
  + Các yếu tố ảnh hưởng đến độ chính xác của mạng neural
  + Quy trình xây dựng mạng neural
* XỬ LÝ ẢNH, TRÍCH CHỌN ĐẶC TRƯNG

# CHƯƠNG 3: NỀN TẢNG PHÁT TRIỂN ỨNG DỤNG NODEJS

## Giới thiệu

CHƯOBG 3: XÂY DỰNG ỨNG DỤNG

* XÂY DỰNG BỘ DỮ LIỆU TRAINING
* MÔ HÌNH NHẬN DẠNG
  + XÂY DỰNG TẬP TRAIN, TẬP TEST
  + CÁCH LÀM, SỐ SL, KÍCH THƯỚC HÌNH
  + PHƯƠNG PHÁP TRÍCH CHỌN ĐẶC TRƯNG
* MẠNG NEURAL
  + ĐẦU VÀO LÀ GÌ
  + SỐ LỚP ẨN
  + SỐ NEURAL LỚP ẨN
  + ĐẦU RA
* XÂY DỰNG ỨNG DỤNG
  + Server side: NodejS
  + Client side: AngularJS, Bootstrap, Jquery,
  + Xử lý ảnh: OpenCV
* KẾT QUẢ

# CHƯƠNG 4: KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN

## Xây dựng mạng neural nhận dạng các ký tự

## Xây dựng ứng dụng

KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN

Kết luận và hướng phát triển…

DANH MỤC TÀI LIỆU THAM KHẢO

1. Yen, Stephen. “NoSQL is a Horseless Carriage” (PDF). NorthScale. Truy cập ngày 26 tháng 6 năm 2014 [↑](#endnote-ref-1)