**ĐẠI HỌC QUỐC GIA TP. HỒ CHÍ MINH**

**TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN**

**KHOA KỸ THUẬT MÁY TÍNH**

**NGUYỄN VĂN PHƯỚC**

**NGUYỄN DUY THỊNH**

**ĐỀ CƯƠNG MÔN HỌC**

**HIỆN THỰC MẠNG NORON TÍCH CHẬP TRÊN FPGA**

**Implementation of noron tich network on FPGA**

**KỸ SƯ KỸ THUẬT MÁY TÍNH**

**TP. HỒ CHÍ MINH, 2020**

**ĐẠI HỌC QUỐC GIA TP. HỒ CHÍ MINH**

**TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN**

**KHOA KỸ THUẬT MÁY TÍNH**

**NGUYỄN VĂN PHƯỚC – 18521263**

**NGUYỂN DUY THỊNH – 18521443**

**ĐỀ CƯƠNG MÔN HỌC**

**HIỆN THỰC MẠNG NORON TÍCH CHẬP TRÊN FPGA**

**Implementation of noron tich network on FPGA**

**KỸ SƯ KỸ THUẬT MÁY TÍNH**

**GIẢNG VIÊN HƯỚNG DẪN**

**TRẦN VĂN CƯƠNG**

**TP. HỒ CHÍ MINH, 2020**

THÔNG TIN HỘI ĐỒNG CHẤM KHÓA LUẬN TỐT NGHIỆP

Hội đồng chấm khóa luận tốt nghiệp, thành lập theo Quyết định số …………………… ngày ………………….. của Hiệu trưởng Trường Đại học Công nghệ Thông tin.

LỜI CẢM ƠN

MỤC LỤC

[Chương 1. TÊN CHƯƠNG 1 3](#_Toc367742496)

[1.1. Chủ đề cấp độ 2 3](#_Toc367742497)

[1.1.1. Chủ đề cấp độ 3 3](#_Toc367742498)

[1.1.2. Chủ đề cấp độ 3 3](#_Toc367742499)

[1.1.2.1. Chủ đề cấp độ 4 3](#_Toc367742500)

[Chương 2. TÊN CHƯƠNG 2 4](#_Toc367742501)

[2.1. Chủ đề cấp độ 2 4](#_Toc367742502)

[2.1.1. Chủ đề cấp độ 3 4](#_Toc367742503)

[2.1.1.1. Chủ đề cấp độ 4 4](#_Toc367742504)

[2.2. Chủ đề cấp độ 2 4](#_Toc367742505)

[2.2.1. Chủ đề cấp độ 3 4](#_Toc367742506)

[Chương 3. TÊN CHƯƠNG 3 5](#_Toc367742507)

[3.1. Chủ đề cấp độ 2 5](#_Toc367742508)

[3.1.1. Chủ đề cấp độ 3 5](#_Toc367742509)

[3.1.1.1. Chủ đề cấp độ 4 5](#_Toc367742510)

[3.2. Chủ đề cấp độ 2 5](#_Toc367742511)

DANH MỤC HÌNH

[Hình 1.1: Tên hình 1 3](#_Toc367742554)

DANH MỤC BẢNG

[Bảng 1.1: Tên bảng 1 3](#_Toc367742567)

[Bảng 2.1: Tên bảng 1 4](#_Toc367742568)

DANH MỤC TỪ VIẾT TẮT

TÓM TẮT ĐỀ CƯƠNG

Chương 1. GIỚI THIỆU TỔNG QUAN

## Mô hình mạng CNN - Convolutional Neural Network

Mạng nơ-ron nhân tạo (Neural Network - NN) là một mô hình lập trình mô phỏng cách thức hoạt động của mạng nơ-ron thần kinh. Kết hợp với các kĩ thuật học sâu (Deep Learning - DL), mạng nơ-ron nhân tạo đang trở thành một công cụ rất mạnh mẽ mang lại hiệu quả tốt nhất cho nhiều bài toán khó như nhận dạng ảnh, giọng nói hay xử lý ngôn ngữ tự nhiên.

Lịch sử phát triển của mạng nơ-ron nhân tạo bắt đầu khi Warren McCulloch và Walter Pitts đã tạo ra một mô hình tính toán cho mạng nơ-ron dựa trên các thuật toán gọi là logic ngưỡng vào năm 1943. Tuy nhiên, để làm quen và hiểu được một số kiến thức cơ bản về mô hình mạng nơ-ron nhân tạp, chúng ta sẽ bàn đến hồi quy logictics - thứ có thể coi là mô hình neural network đơn giản nhất với chỉ input layer và output layer.

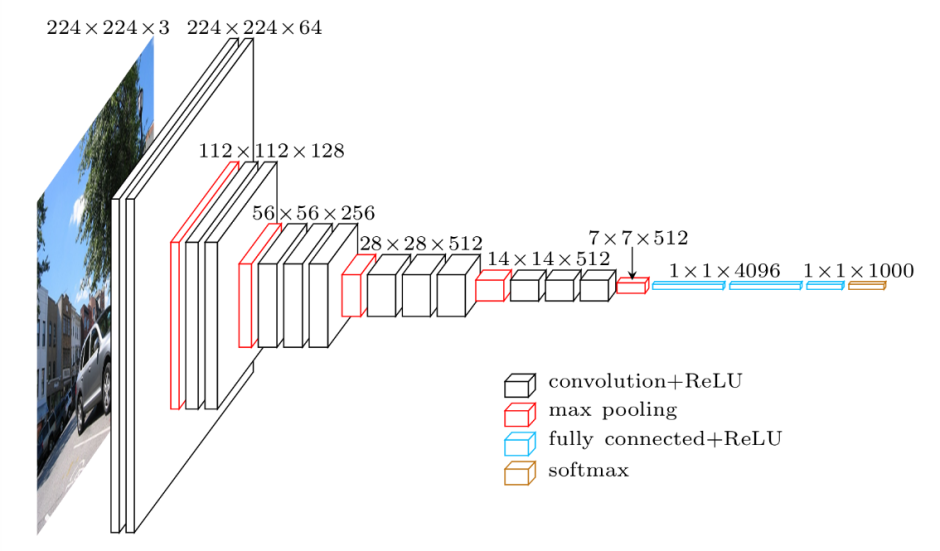
Mạng nơron tích chập (còn gọi là ConvNet / CNN) là một thuật toán Deep Learning có thể lấy hình ảnh đầu vào, gán độ quan trọng (các trọng số - weights và độ lệch - bias có thể học được) cho các đặc trưng/đối tượng khác nhau trong hình ảnh và có thể phân biệt được từng đặc trưng/đối tượng này với nhau. Công việc tiền xử lý được yêu cầu cho mạng nơron tích chập thì ít hơn nhiều so với các thuật toán phân loại khác. Trong các phương thức sơ khai, các bộ lọc được thiết kế bằng tay (hand - engineered), với một quá trình huấn luyện để chọn ra các bộ lọc/đặc trưng phù hợp thì mạng nơron tích chập lại có khả năng tự học để chọn ra các bộ lọc/ đặc trưng tối ưu nhất.

Kiến trúc của nơron tích chập tương tự như mô hình kết nối của các nơron trong bộ não con người và được lấy cảm hứng từ hệ thống vỏ thị giác trong bộ não (visual cortex). Các nơ-ron riêng lẻ chỉ phản ứng với các kích thích trong một khu vực hạn chế của trường thị giác được gọi là Trường tiếp nhận (Receptive Field). Một tập hợp các trường như vậy chồng lên nhau để bao phủ toàn bộ khu vực thị giác.

## Mô hình mạng CNN VGG-19

Khi nhắc đến VGG19 chúng ta nên nhắc lại về tiền thân của nó – VGG-16. Được phát triển năm 2014, là một biến thể sâu hơn nhưng lại đơn giản hơn so với kiến trúc convolution (từ gốc: convolutional structure) thường thấy ở CNN Kiến trúc như hình dưới, có thể thấy số mặc dù các tầng cao hơn được đơn giản hóa so với LeNet, AlexNet … thu gọn về kích thước nhưng số lượng lại lớn hơn, sâu hơn

Số lượng tham số: 138.000.000

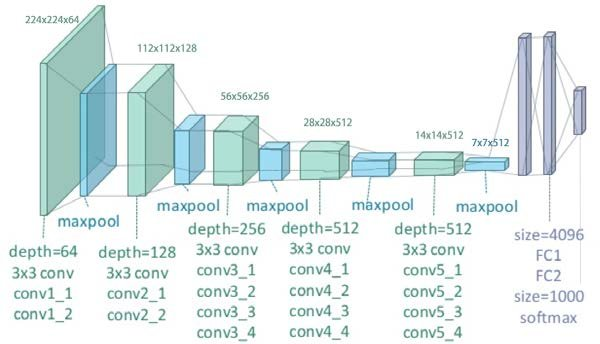


Hình 1.2.1 Kiến trúc mạng VGG-16

Với VGG-16, quan điểm về một mạng nơ ron sâu hơn sẽ giúp ích cho cải thiện độ chính xác của mô hình tốt hơn. Về kiến trúc thì VGG-16 vẫn dữ các đặc điểm của AlexNet nhưng có những cải tiến:

* Kiến trúc VGG-16 sâu hơn, bao gồm 13 layers tích chập 2 chiều (thay vì 5 so với AlexNet) và 2 layers fully connected. Đối với VGG-19, sẽ gồm 16 layers tích chập (mỗi conv tăng 1 bộ với 3 conv cuối) và tăng 1 layers fully connected.
* Lần đầu tiên trong VGG-16 chúng ta xuất hiện khái niệm về khối tích chập (block). Đây là những kiến trúc gồm một tập hợp các layers CNN được lặp lại giống nhau. Kiến trúc khối đã khởi nguồn cho một dạng kiến trúc hình mẫu rất thường gặp ở các mạng CNN kể từ đó.
* VGG-16 cũng kế thừa lại hàm activation ReLU ở AlexNet.
* VGG-16 cũng là kiến trúc đầu tiên thay đổi thứ tự của các block khi xếp nhiều layers CNN + max pooling thay vì xen kẽ chỉ một layer CNN + max pooling. Một bạn có câu hỏi tại Forum Machine Learning Cơ Bản về sự thay đổi này sẽ giúp cho VGG net cải thiện hơn như thế nào? Các layers CNN sâu hơn có thể trích lọc đặc trưng tốt hơn so với chỉ 1 layers CNN.
* VGG-16 chỉ sử dụng các bộ lọc kích thước nhỏ 3x3 thay vì nhiều kích thước bộ lọc như AlexNet. Kích thước bộ lọc nhỏ sẽ giúp giảm số lượng tham số cho mô hình và mang lại hiệu quả tính toán hơn. VD: Nếu sử dụng 2 bộ lọc kích thước 3 x 3 trên một featurs map (là output của một layer CNN) có độ sâu là 3 thì ta sẽ cần n\_filters x kernel\_size x kernel\_size x n\_channels = 2 x 3 x 3 x 3 = 54 tham số. Nhưng nếu sử dụng 1 bộ lọc kích thước 5 x 5 sẽ cần 5 x 5 x 3 = 75 tham số. 2 bộ lọc 3 x 3 vẫn mang lại hiệu quả hơn so với 1 bộ lọc 5 x 5.
* Mạng VGG-16 sâu hơn so với AlexNet và số lượng tham số của nó lên tới 138 triệu tham số. Đây là một trong những mạng mà có số lượng tham số lớn nhất. Kết quả của nó hiện đang xếp thứ 2 trên bộ dữ liệu ImageNet validation ở thời điểm public. Ngoài ra còn một phiên bản nữa của VGG-16 là VGG-19 tăng cường thêm 3 layers về độ sâu.

Bắt đầu từ VGG-16, một hình mẫu chung cho các mạng CNN trong các tác vụ học có giám sát trong xử lý ảnh đã bắt đầu hình thành đó là các mạng trở nên sâu hơn.



Hình 1.2.2 Kiến trúc mạng VGG-19

## Mục tiêu của đề tài

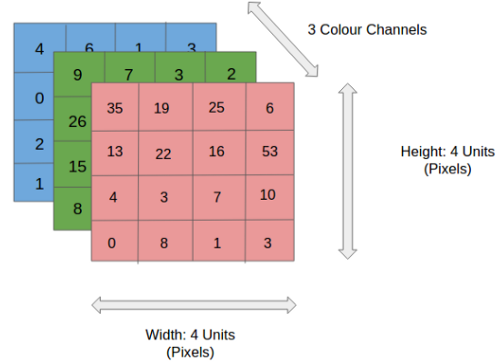
Trong môn học này, đầu tiên, nhóm sẽ nghiên cứu, tìm hiểu tổng quan về mạng CNN và các thành phần bên trong. Sau đó, nhóm sẽ nghiên cứu và phân tích mạng CNN cụ thể, đó là VGG-19 và ứng dụng nó vào phân loại chó mèo, sử dụng ngôn ngữ Verilog để thực thi thiết kế để chạy được trên Kit FPGA với đầu vào (input) là một tập ảnh về dụng cụ học tập (bút,thước, tẩy) và output của nhóm mong đợi là phân loại hình nào là bút, hình nào là thước, hình nào là tẩy.

# THÀNH PHẦN CỦA MẠNG CNN

## Convolutional Layer – Lớp tích chập

Đầu vào của Convnets là một bức ảnh được biểu diển bởi ma trận pixel với kích thước: [w x h x d]

* W: chiều rộng
* H: chiều cao
* D: Là độ sâu, hay dễ hiểu là số lớp màu của ảnh. Ví dụ ảnh RBG sẽ là 3 lớp ảnh Red, Green, Blue

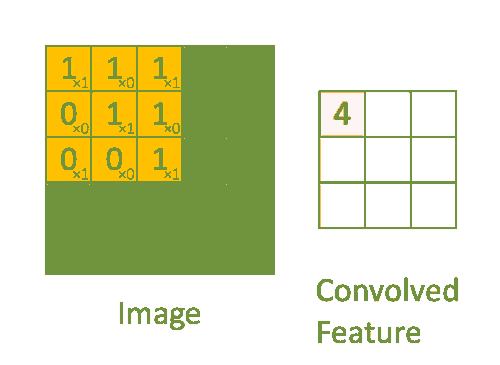


Hình 2.2.1 Một hình RGB có kích thước 4x4x3

Kích thước hình ảnh = 5 (Chiều cao) x 5 (Chiều rộng) x 1 (Số lượng kênh, ví dụ: RGB).

Ở hình bên, phần màu xanh lục là hình ảnh đầu vào 5×5×1 của chúng ta, ta gọi hình ảnh đầu vào này là I

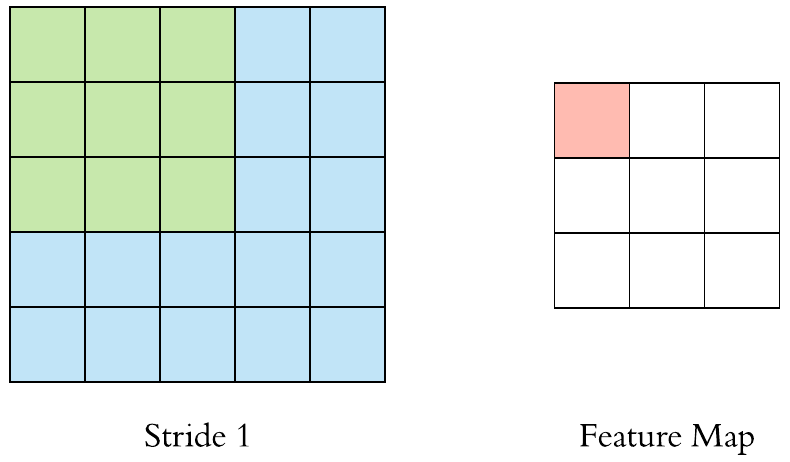
Phần tử liên quan đến việc thực hiện thao tác tích chập trong phần đầu tiên của lớp tích chập được gọi là **Bộ lọc (Kernel / Filter)**, K, được thể hiện bằng màu vàng. Chúng ta chọn K là một ma trận 3x3x1



Hình 2.2 Tích chập một hình ảnh 5x5x1 với một bộ lọc 3x3x1 để có được một hình ảnh chập 3x3x1

### Stride và Padding

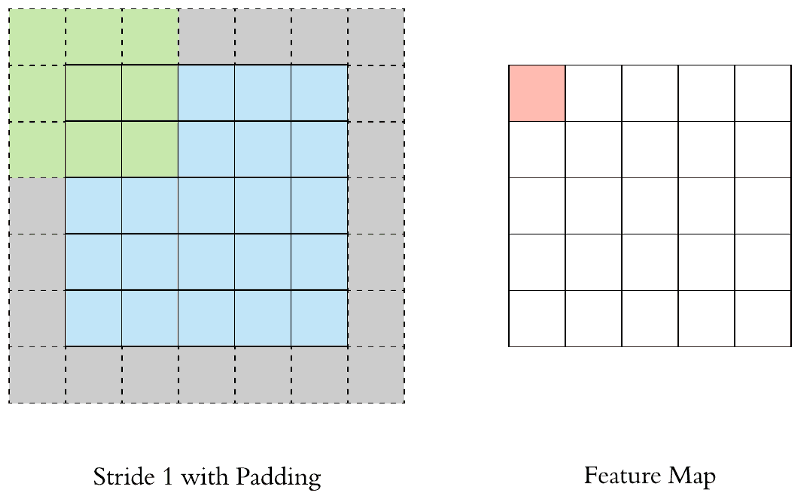
**Stride** là khoảng cách giữa 2 **kernel** khi quét. Với stride = 1, **kernel** sẽ quét 2 ô ngay cạnh nhau, nhưng với stride = 2, **kernel** sẽ quét ô số 1 và ô số 3. Bỏ qua ô ở giữa. Điều này nhằm tránh việc lặp lại giá trị ở các ô bị quét. Stride = 1.



Hình 2.3 Thanh trượt của lớp Conv với stride = 1

Chúng ta chọn stride và size của kernel càng lớn thì size của feature map càng nhỏ, một phần lý do đó là bởi kernel phải nằm hoàn toàn trong input. Có một cách để giữ nguyên kích cỡ của feature map so với ban đầu. Đấy là Padding. Khi ta điều chỉnh padding = 1, tức là ta đã thêm 1 ô bọc xung quanh các cạnh của input, muốn phần bọc này càng dày thì ta cần phải tăng padding lên.

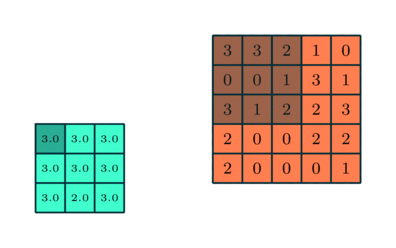
Hãy nhìn vào ví dụ sau, ta xét padding = 1:



Hình 2.4 Thanh trượt với Stride =1 và Padding =1

## Pooling Layer – Lớp tổng hợp

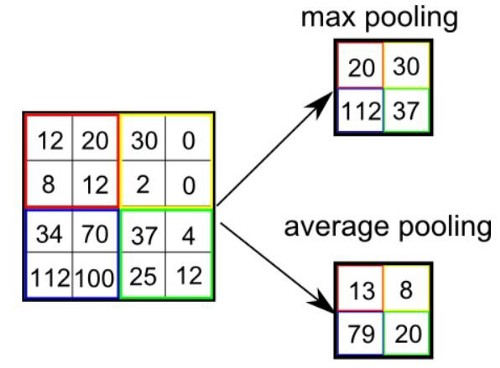
Tương tự như lớp tích chập (Convolutional Layer), lớp gộp (pooling) chịu trách nhiệm để là giảm chiều kết quả tích chập (Convolved Feature). Điều này nhằm mục đích để giảm chi phí tính toán cần phải có để xử lý dữ liệu thông qua việc giảm kích thước tính năng đầu vào. Hơn nữa, nó rất hữu ích để trích xuất các đặc trưng cốt lõi, cái thường bất biến trước các phép xoay và phép trượt, do đó làm cho quá trình huấn luyện mô hình hiệu quả hơn.



Hình 2.5 Ma trận gộp 3x3 từ một ma trận tích chập 5x5

Có hai loại phép gộp: Gộp cực đại (Max Pooling) và Gộp trung bình (Average Pooling). Phép gộp cực đại trả về giá trị lớn nhất từ phần hình ảnh được bao phủ bởi bộ lọc. Trong khi đó, phép gộp trung bình trả về giá trị trung bình của tất cả các giá trị từ phần hình ảnh được bao phủ bởi bộ lọc.

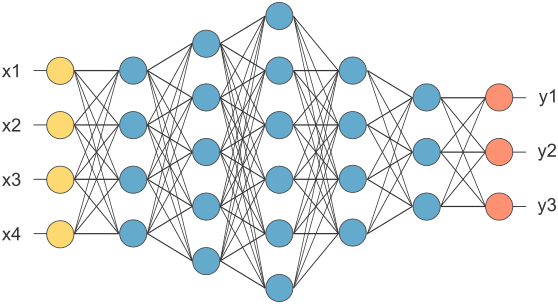
Phép gộp cực đại cũng hoạt động như một công cụ khử nhiễu. Nó loại bỏ các nguồn nhiễu và thực hiện khử nhiễu song song với giảm kích thước. Mặt khác, phép gộp trung bình chỉ đơn giản thực hiện giảm kích thước như một cơ chế khử nhiễu. Do đó, chúng ta có thể nói rằng phép gộp cực đại hoạt động tốt hơn rất nhiều so với phép gộp trung bình.



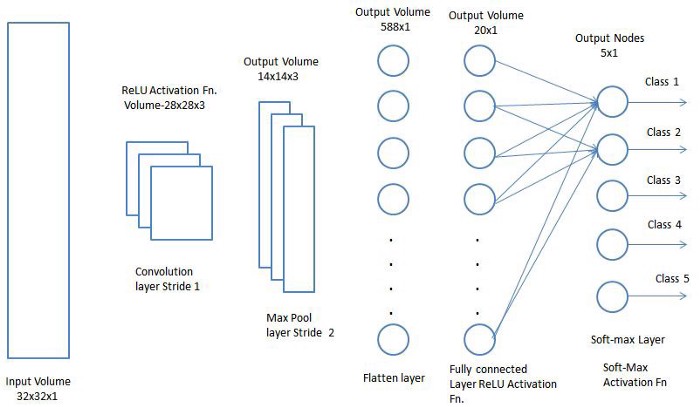
Hình 2.6 Các kiểu phép gộp

## Fully Connect Layer – Lớp kết nối đầy đủ

Sau khi ảnh được truyền qua nhiều convolutional layer và pooling layer thì model đã học được tương đối các đặc điểm của ảnh thì tensor của output của layer cuối cùng sẽ được là phẳng thành vector và đưa vào một lớp được kết nối như một mạng nơ-ron. Với FC layer được kết hợp với các tính năng lại với nhau để tạo ra một mô hình. Cuối cùng sử dụng softmax hoặc sigmoid để phân loại đầu ra.



Hình 2.7 Ví dụ về lớp FC



Hình 2.8 Ví dụ về lớp FC

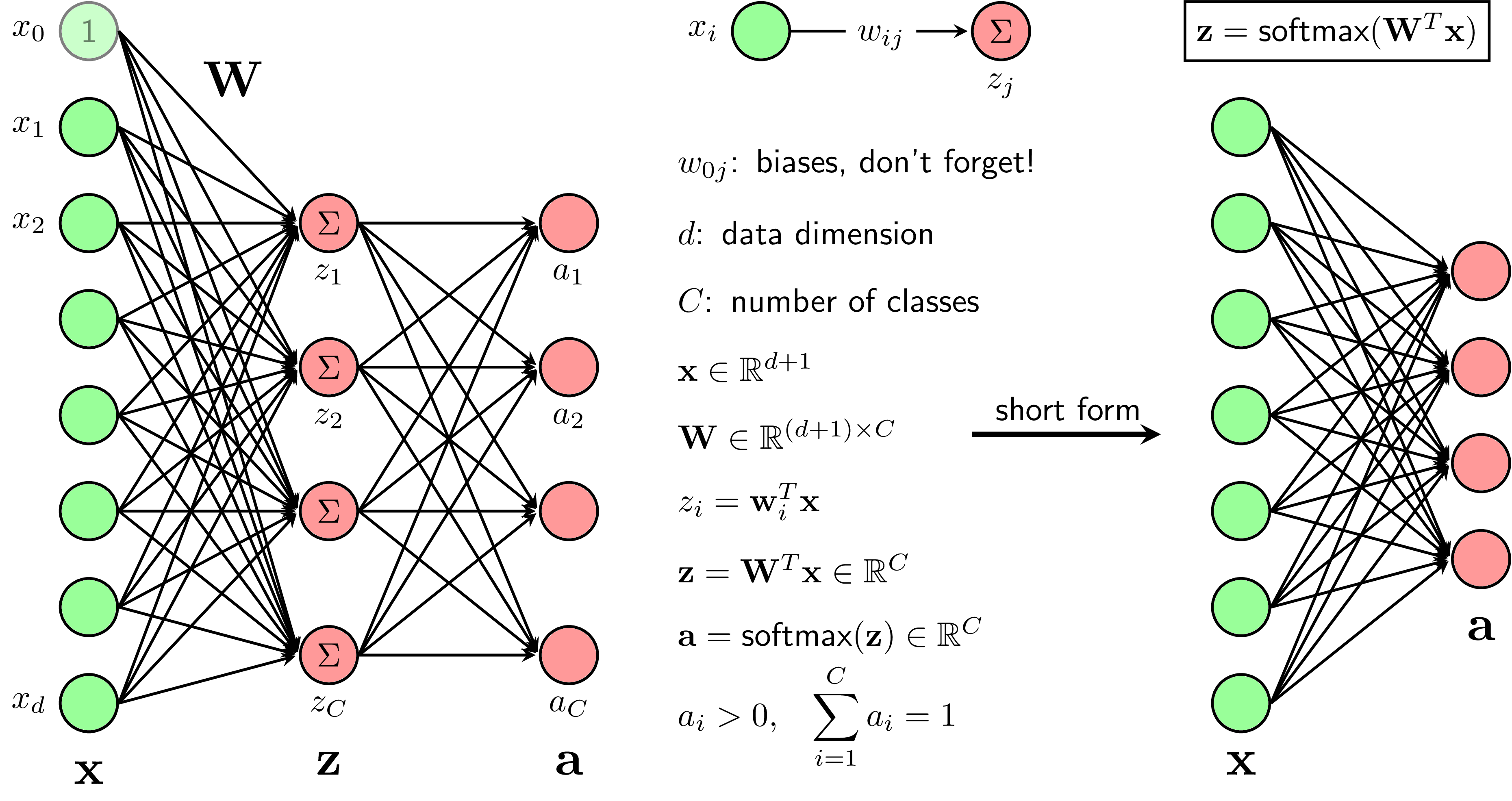
## Các hàm kích hoạt thường dùng

### ReLu

Hàm ReLU đang được sử dụng khá nhiều trong những năm gần đây khi huấn luyện các mạng neuron. ReLU đơn giản lọc các giá trị < 0. Nhìn vào công thức chúng ta dễ dàng hiểu được cách hoạt động của nó.

f(x) = max (0, x)

### Softmax



Hình 2.9 Mô hình Softmax Regression dưới dạng Neural network

Ở phần bên phải, hàm tuyến tính Σvà hàm softmax (activation function) được tách riêng ra để phục vụ cho mục đích minh họa. Dạng short form ở bên phải là dạng hay được sử dụng trong các Neural Networks, lớp a được ngầm hiểu là bao gồm cả lớp z.

Hàm kích hoạt Softmax thường được dùng sau khi thực hiện FC Layer, mục đích để giá trị được chuẩn hoá dưới dạng xác suất, hình trên mô tả lớp kết nối đầy đủ và theo sau là hàm kích hoạt Softmax. Với công thức tổng quát của hàm kích hoạt Softmax (là xác suất dự đoán của từng lớp được thể hiện như công thức (1)

(1)

TÀI LIỆU THAM KHẢO

Theo chuẩn IEEE