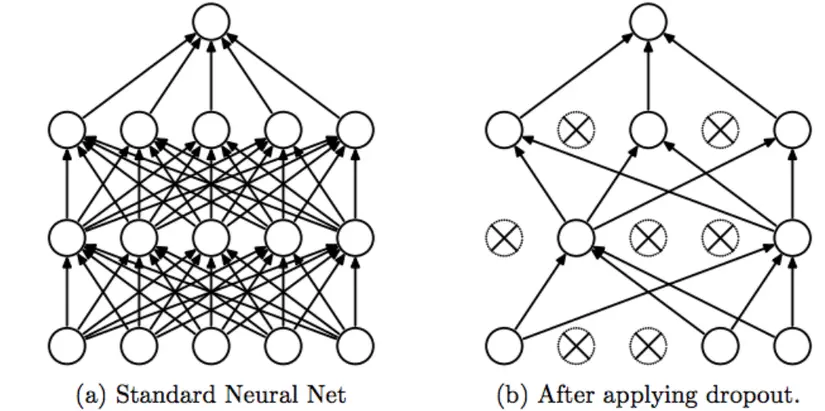
**Phần 2\_5\_Lớp Dropout**

### **I. Nội dung chính**

Về các phép tính toán (lớp tính toán) để xây dựng mô hình học sâu.

* + (5) Lớp **Dropout**
    - Nguyên lý làm việc
    - Độ rộng của vùng nhận thức (receptive fields)
    - Hình dạng kích thước của bản đồ đặc trưng đầu vào và đầu ra

### **II. Nội dung biên soạn chi tiết**



1 Dropout trong mạng Neural là gì?

Hiểu 1 cách đơn giản thì Dropout là việc **bỏ qua các đơn vị (tức là 1 nút mạng)** trong quá trình đào tạo 1 cách **ngẫu nhiên**. Bằng việc bỏ qua này thì đơn vị đó sẽ **không được xem xét trong quá trình forward và backward**. Theo đó, p được gọi là xác suất giữ lại 1 nút mạng trong mỗi giai đoạn huấn luyện, vì thế xác suất nó bị loại bỏ là **(1 - p)**.

* Quá trình **training**: Đối với mỗi lớp ẩn, mỗi example, mỗi vòng lặp, ta sẽ **bỏ học 1 cách ngẫu nhiên với xác suất (1 - p) cho mỗi nút mạng**.
* Quá trình **test**: Sử dụng **tất cả các kích hoạt.**

**Code example:**

**dropout1 = 0.2 , dropout2 = 0.5**

**net = nn.Sequential()**

**net.add(nn.Dense(256, activation="relu"),**

***# Add a dropout layer after the first fully connected layer***

**nn.Dropout(dropout1),**

**nn.Dense(256, activation="relu"),**

***# Add a dropout layer after the second fully connected layer***

**nn.Dropout(dropout2),**

**nn.Dense(10))**

**Tại sao lại cần Dropout**

Tại sao phải tắt 1 số nút mạng theo đúng nghĩa đen trong quá trình huấn luyện ? Câu trả lời là: **Tránh học tủ (Over-fitting)**

Nếu 1 lớp fully connected có quá nhiều tham số và chiếm hầu hết tham số, các nút mạng trong lớp đó quá phụ thuộc lẫn nhau trong quá trình huấn luyện thì sẽ hạn chế sức mạnh của mỗi nút, dẫn đến việc kết hợp quá mức.

Dropout sẽ được **học thêm các tính năng** mạnh mẽ hữu ích

Nó gần như **tăng gấp đôi số epochs cần thiết để hội tụ**. Tuy nhiên, **thời gian cho mỗi epoch là ít hơn**.

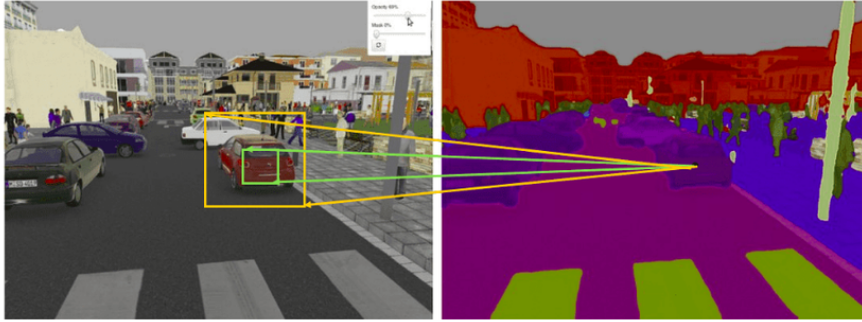
Ta có **H đơn vị ẩn**, với xác suất bỏ học cho mỗi đơn vị là **(1 - p)** thì ta có thể có **2^H mô hình** có thể có. Nhưng trong giai đoạn test, tất cả các nút mạng phải được xét đến.

**2. Receptive field:**

Receptive field trong deep learning được định nghĩa là kích thước của một **vùng (region) trong không gian đầu vào (input space)** được nhìn thấy bởi pixel output qua một kernel/filter.

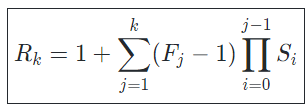
**Tại sao lại cần receptive field:**

Để dễ hình dung ta sẽ lấy một ví dụ cụ thể, như chúng ta đã biết trong bài toán image segmentation nhiệm vụ là đi dự đoán từng pixel tương ứng ở input xem nó đang thuộc class tương ứng nào. Vậy điều chúng ta mong muốn nhất ở đây sẽ là làm sao cho việc dự đoán pixel đó thuộc true class là cao nhất. Mà điều kiện lý tưởng nhất cho điều này là tại mỗi output pixel nó có một **receptive field thật lớn**. Điều này giúp cho việc model không bị bỏ qua những chi tiết quan trọng trong quá trình dự đoán. Như bạn thấy ở hình bên dưới, với ouput pixel bạn muốn dự đoán thuộc khu vực chiếc ô tô thì bạn muốn receptive field là màu cam hay màu xanh? Options nào sẽ mang lại nhiều thông tin cho output hơn?

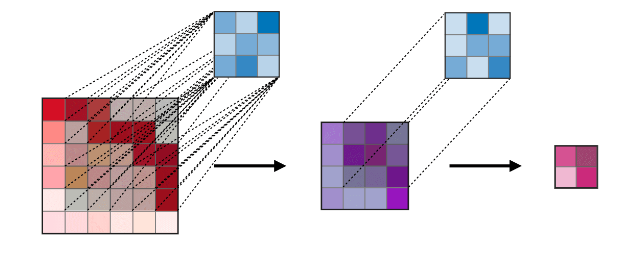


**Trường thụ cảm** Trường thụ cảm (receptive field) tại tầng *k* là vùng được ký hiệu

***Rk* × *Rk*** của đầu vào mà những **pixel của activation map thứ *k* có thể "nhìn thấy".** Bằng cách gọi *Fj* là kích thước bộ lọc của tầng *j* và *Si* là giá trị độ trượt của tầng i và để thuận tiện, ta mặc định *S*0=1, trường thụ cảm của tầng *k* được tính toán bằng công thức:



*Trong ví dụ bên dưới, ta có* *F*1=*F*2=3 *và S*1=*S*2=1*, nên cho ra được R*2=1+2⋅1+2⋅1=5*.*

**