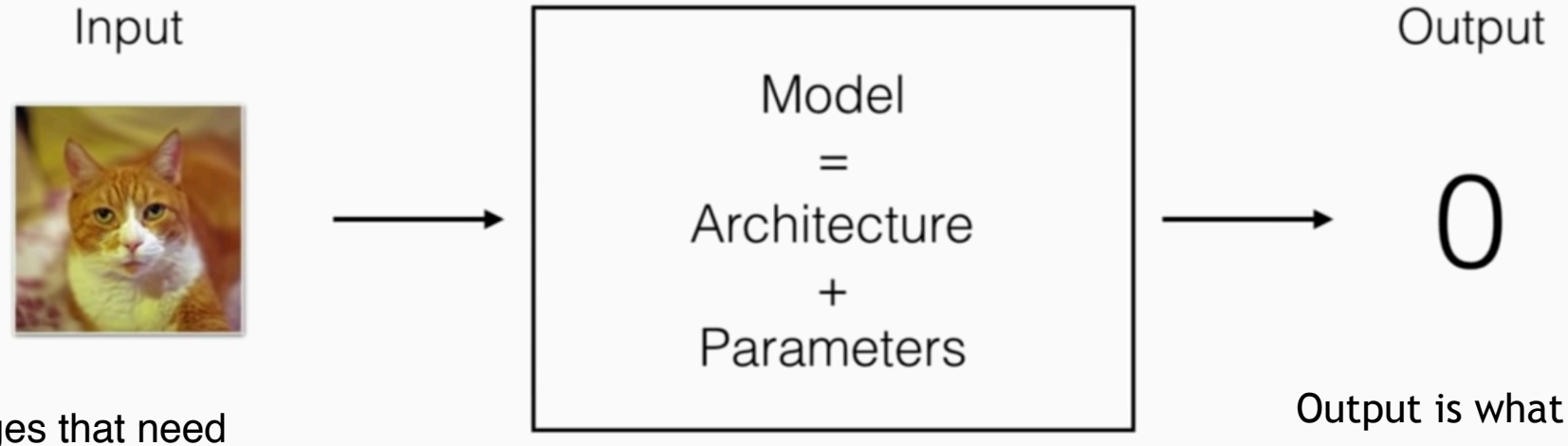
Outline video:
Lecture 2 - Deep Learning Intuition

Trần Ngọc Bảo Duy - 51702091

1. What will we do in class ?

- How to label the data properly, and achieve high results in training and evaluation through practice.
- How to choose a model architecture that suits your data
- The optimal way to design the loss function provides good results

2. Model Neural Network



- Inputs are images that need detect, such as a cat image, a recording, an article

- Architecture is a model architecture: logistic regression, neural network, CNN,
- Parameters: parameters that match the model to make the model better - the loss function will decrease faster, the rest of the code such as the cat handling function to numpy, word to vec,

Output is what we want

Such as classify, multi-classify, detect object, ...

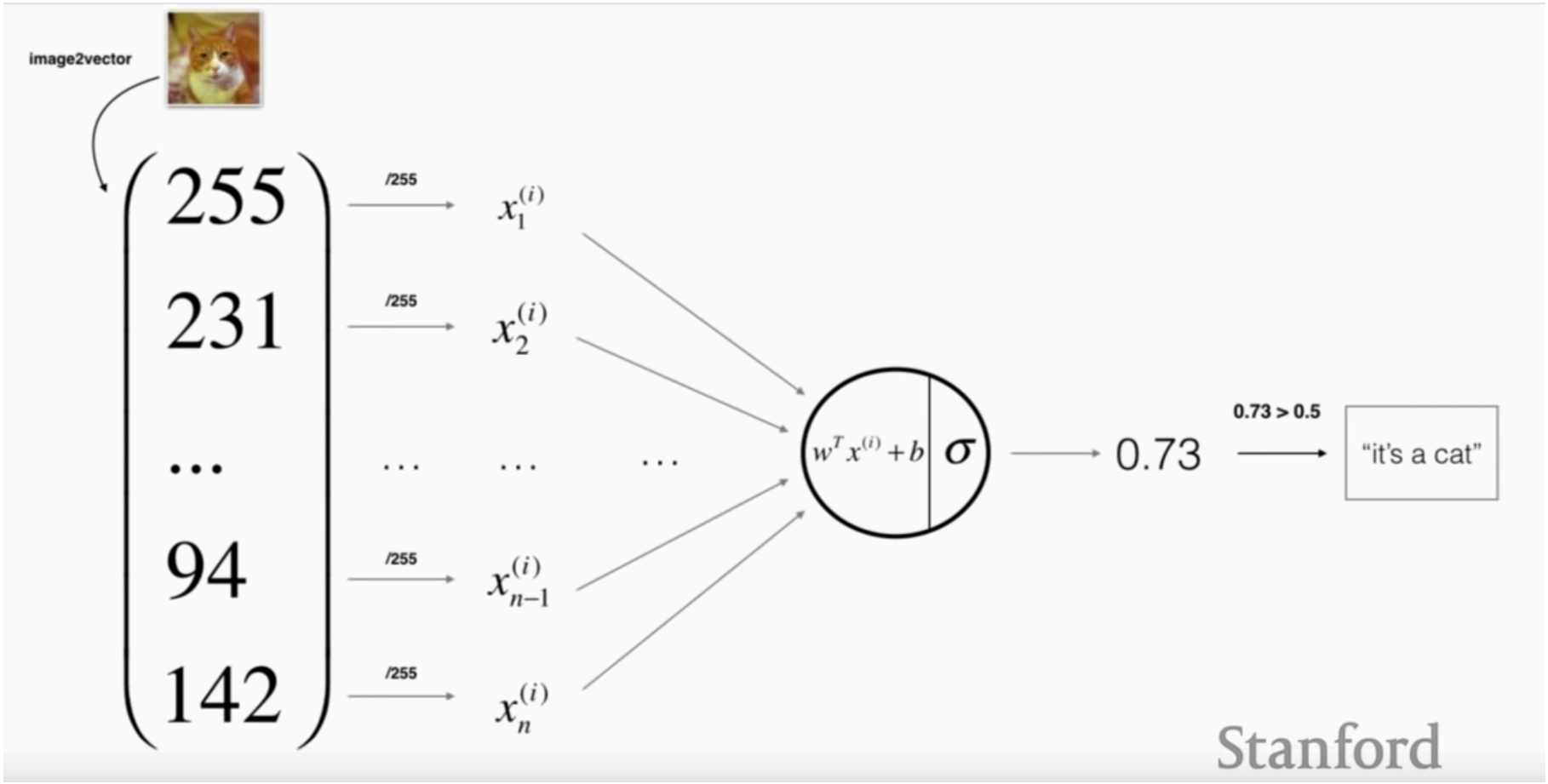
as shown above - classify:

0 pose is no cat

1 pose cat in the picture

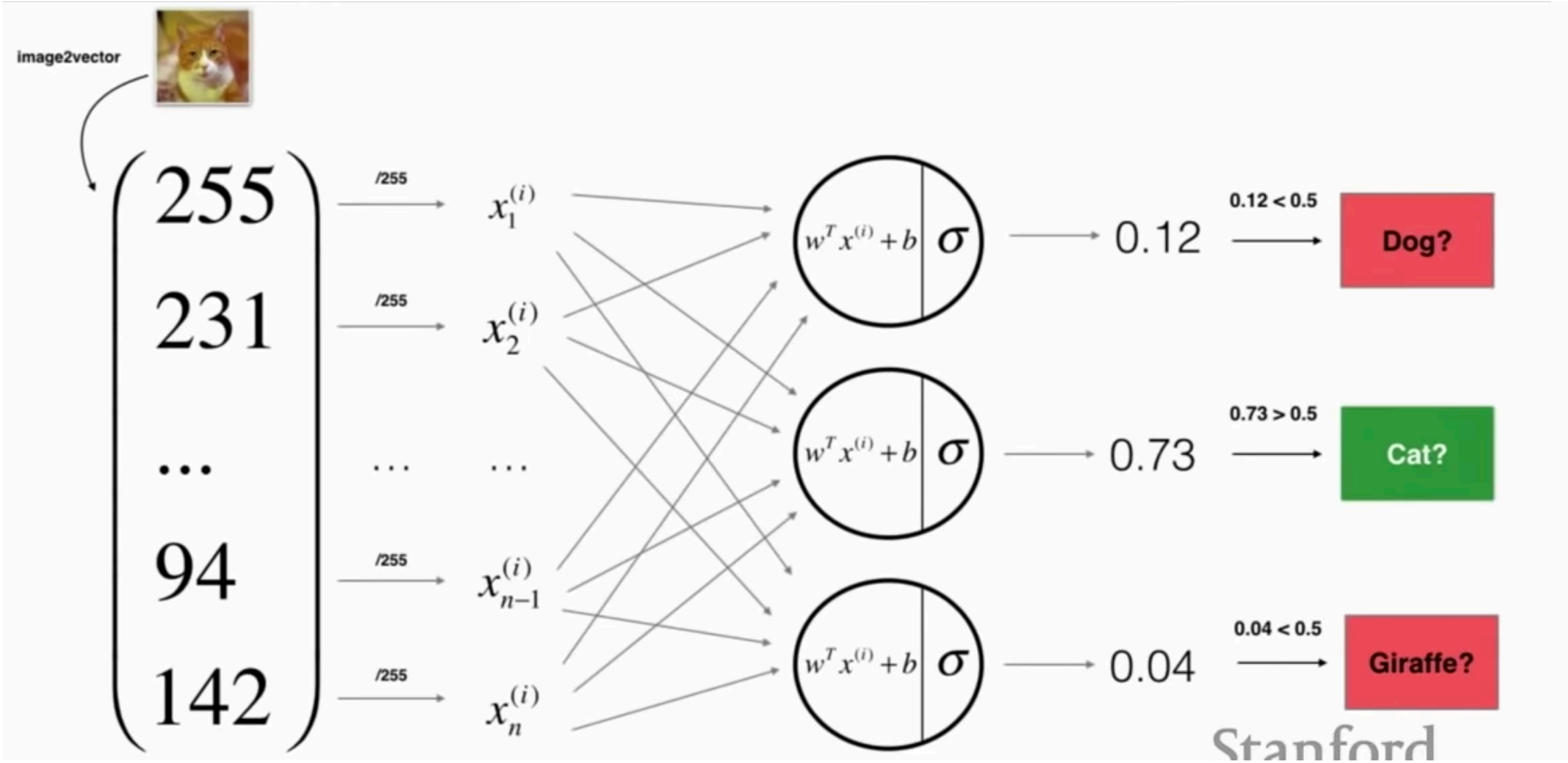
2. Model Neural Network

- Have or no have, use sigmoid



2. Model Neural Network

- Multi class classification: use softmax function instead of sigmoid to produce results of all classes



2. Model Neural Network

- In this lecture we will learn about:

- Analyze the problem from the perspective of a deep learning engineer
- How to choose a model architecture
- How to choose loss function, training strategy, activation function

2. Day - Night Classification

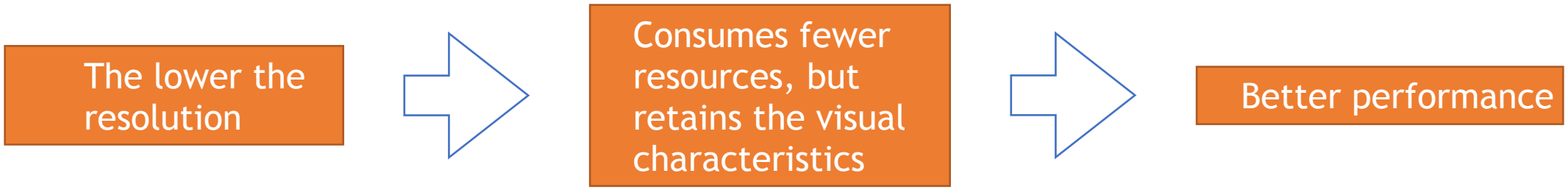
- **Data**

  - Collect data: 10000 images, if you want to be good, must have more photos
  - Classification day and night, indoors or outdoors, dawn, sunrise, sunset, dusk, ...
  - Train, test split: usually divide 80/20 if the data is bigger we can increase train set to 90% or even higher.
  - Bias: balance the images in each class

2. Day - Night Classification

- **Input**

  - The pixel of the input image
  - Image resolution: 32x32 will be different from 400x400x3 because of RGB color images
  - Good resolution according to 64x64x3 follow video, because there are some very similar images, if the image is too small will not distinguish, may be larger



2. Day - Night Classification

- **Output**

  - We specify that night is 0 and day is 1, output will be y = 0 l y =1
  - What is last activation function? Because it is either 0 or 1 => it is a Sigmoid activation function

- **Architecture**

  - Don’t need use deep network, may be use shallow network because it do well in this case

- **Loss**

  - Use log-likelihood L = -[y*log(y^) + (1-y)*log(1-y^)]

3. Face - Verification

- **Data**

  - Picture of student with labelled their name

- **Input**

  - The pixel of the input image: 400x400x3, because we use video to detect, so maybe there were parts is harder to detect. That's reason why in the problem classify cat, we just use dimension 64x64x3
  - Maybe use color or none color, because color in problem not a feature, but in the problem day - night the color is important parameter
  - The more complex task, the more data we will need

3. Face - Verification

- **Output**

- Y = 1 is you or y = 0 is not you

- **Architecture**

- Don't use this way

- Simple solution (tradition): we will compute distance pixel per pixel. Put out 1 threshold: if output < threshold => y = 1

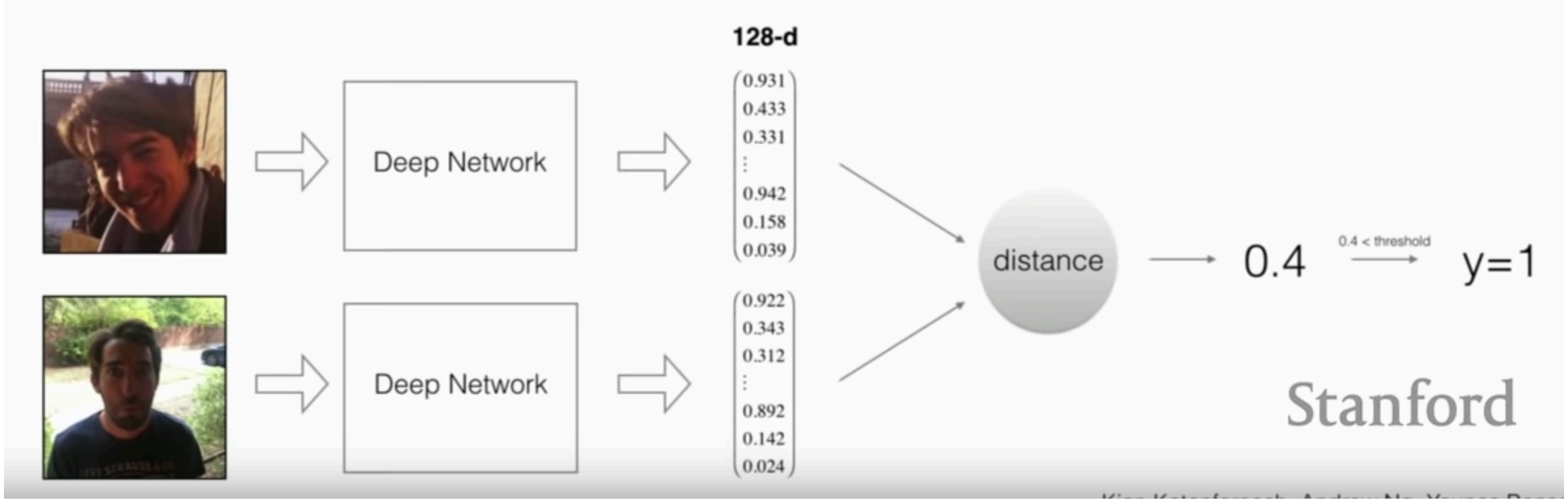
=> This is normal way so far, because the result maybe noise

- Issues:
  - ➡ Background lighting differences
  - ➡ A person can wear makeup, grow a-bread
  - ➡ Id photo can be outdated

3. Face - Verification

- **<u>Architecture</u>**

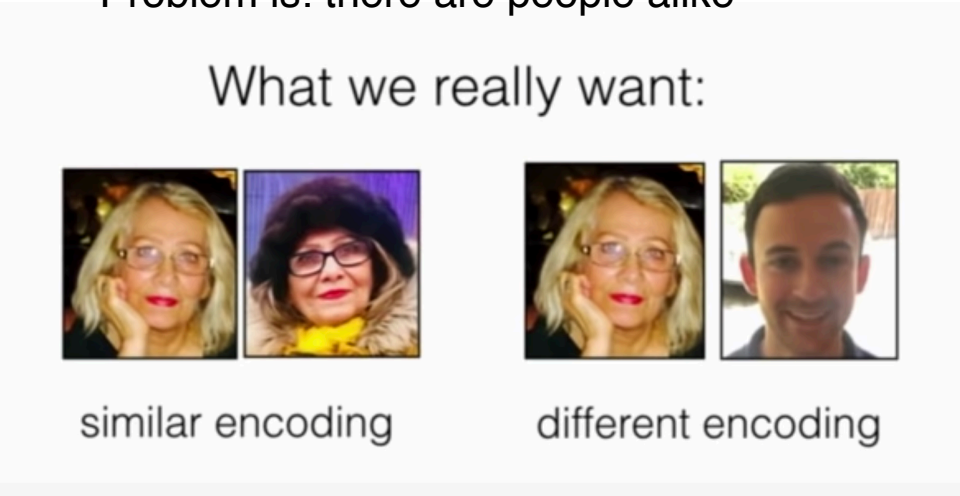
  - Our solution: encode information about picture in a vector
  - We gather all students face encoding in a db. When we have a new image, we compute it with the encoded one

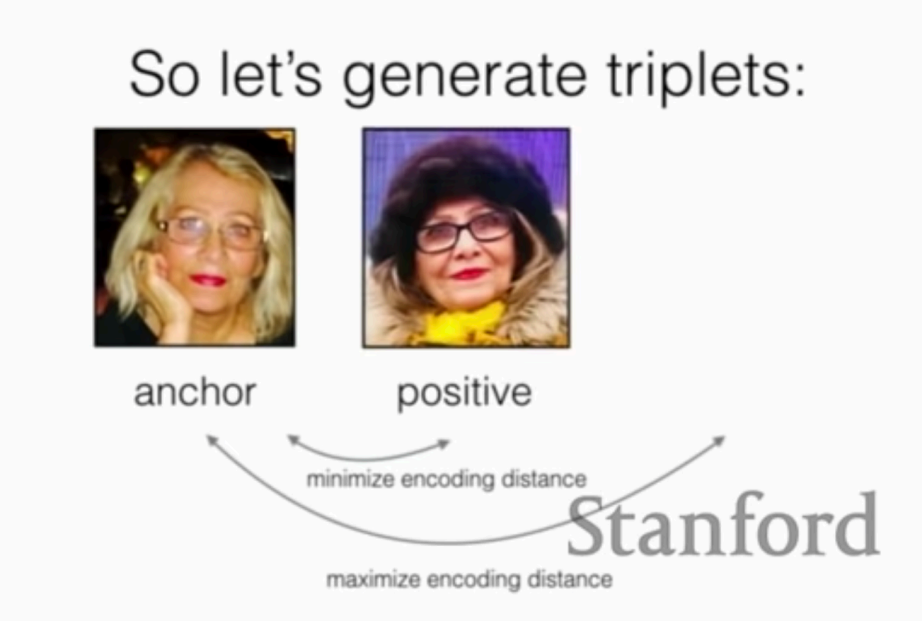


3. Face - Verification

- **Train**

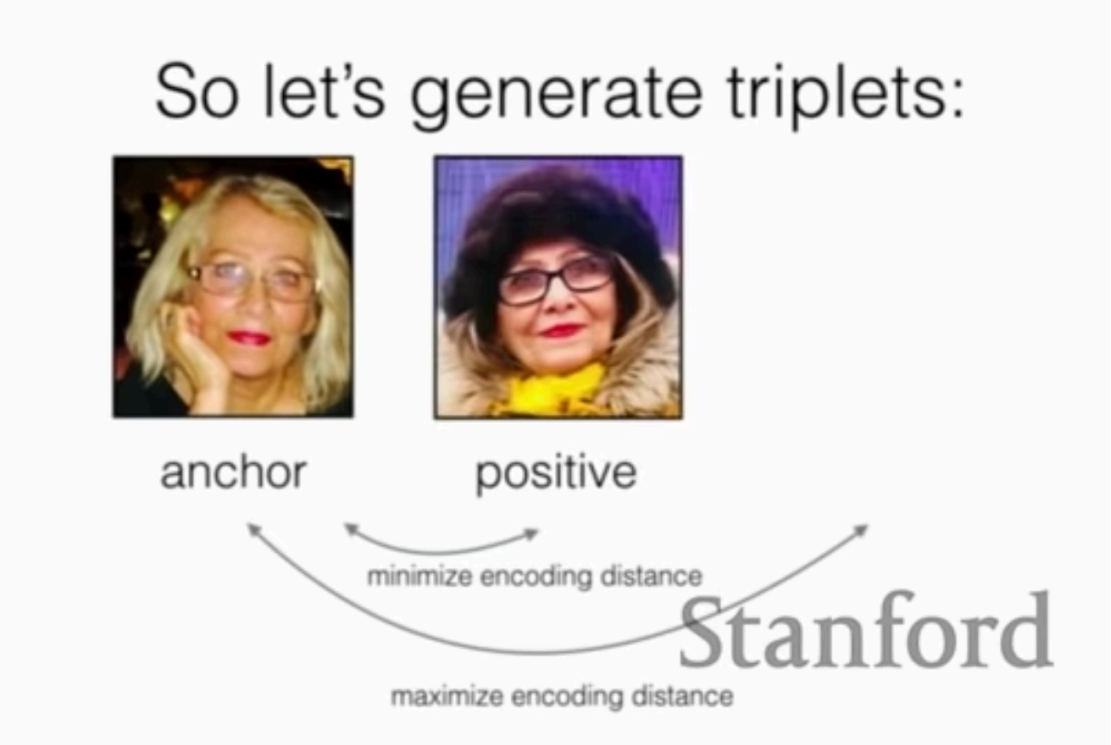
  - To understand how to encode: we should be get more data
  - Because every year will have many new students -> dataset maybe more larger, we don’t want retrain model
  - Problem is: there are people alike





3. Face - Verification

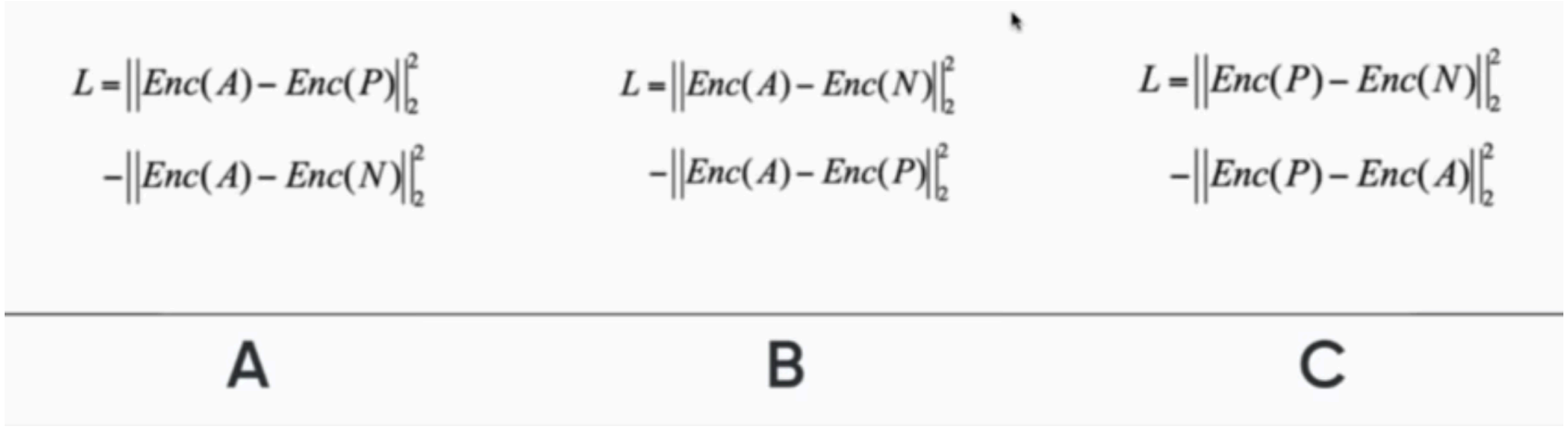
- **Train**



- We must define loss: We want to do
  - minimize the encoding distance between the anchor and the positive
  - maximize the encoding distance between the anchor and the negative

3. Face - Verification

- **Loss**
  - We have three loss functions here.
  - We want minimize the encoding distance between the anchor and the positive and maximize the encoding distance between the anchor and the negative so we must subtract (A-P) - (A-N) => ***choose A***
  - ***Note: we must have α because we don't want Enc(A) = Enc(P) = Enc(N) => loss 0 ! We don't do that happend so we add a α in after the equation Loss***



3. Face - Verification

- **Train**

  - Solution Model: Simple, we train a model to crop a face in quite big picture and then detect that one
  - Or we can detect at one time by way: we have information for all students in db, at this time we detect we compare to all that. => We cannot predict at the same time, because it took many space in storage

  ➡ ***We Use K-Nearest Neighbors***

  - We want to group picture of same people => K-Mean Algorithm. How to define a K number? Define them by trainer clustering algorithm and look a certain loss and you define how small it is
  - Using X-means to find the K - Elbow method

4. Art Generation

- Given a picture and make it's more beautiful

- **Data**
  - Any data what we want.
- **Input**

content image

style image

- **Output**

generated image

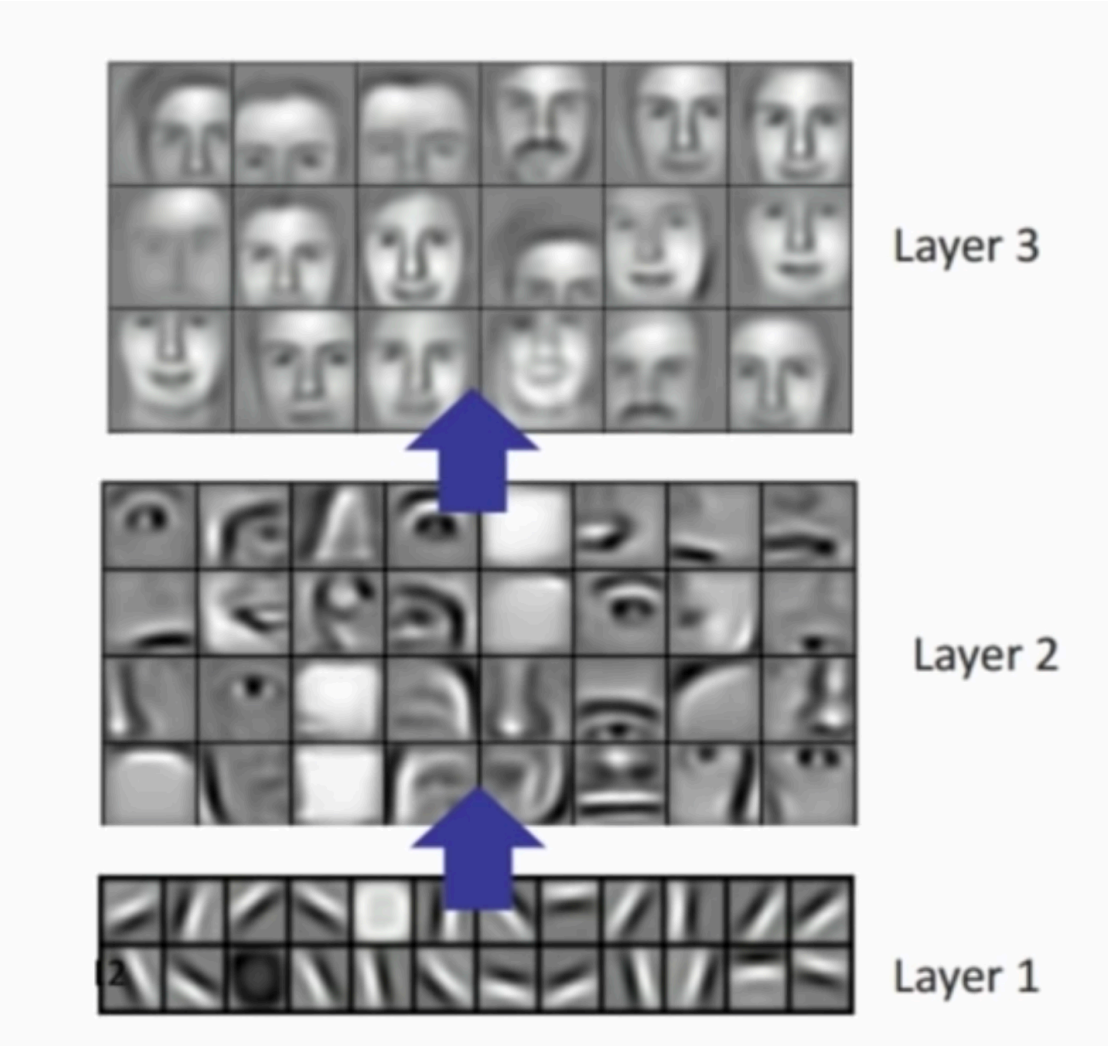
- A generative image include content of content image and decorate by style image

4. Art Generation

- **Architecture**

◉ **Define the content image and style image:**

- The content image: we can detect the edges in the earlier layer, because the edges are usually a good representation of the content image, extract the information from first layer
- The style image: is non-localized information. It's a technique call Gram matrix



4. Art Generation

- **Architecture**

◉ **Define the content image and style image:**

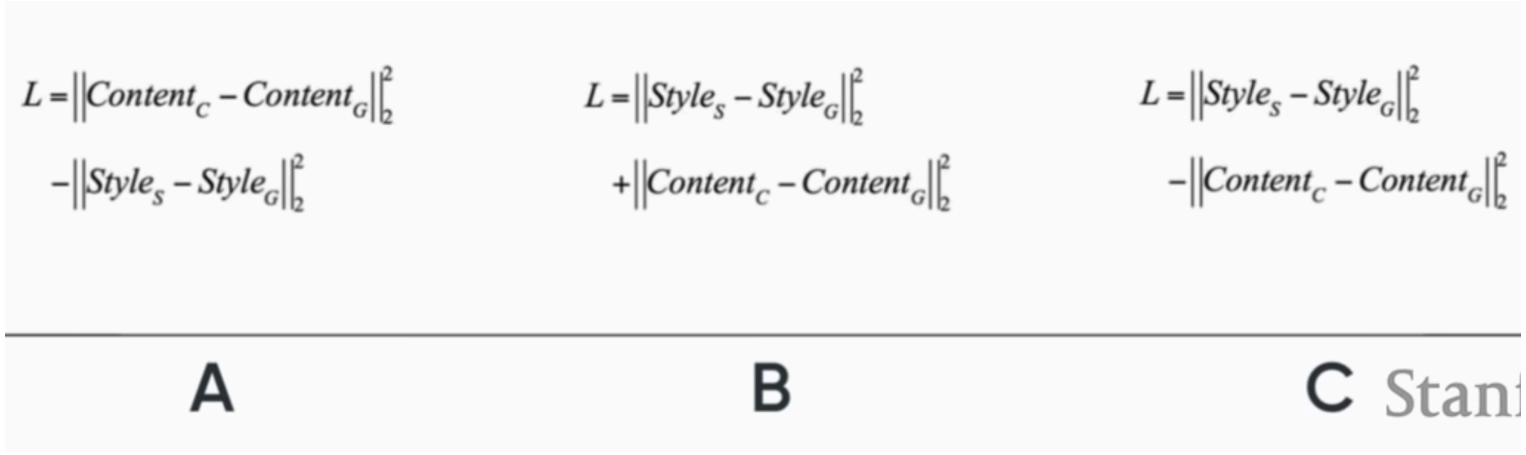
- We use data of ImageNet because it were trained to recognize more than thousand of thousands of objects. (train in months, with thousand classes)
- We can get feature of content and style image we call

$$Content_C$$
$$Style_S$$

4. Art Generation

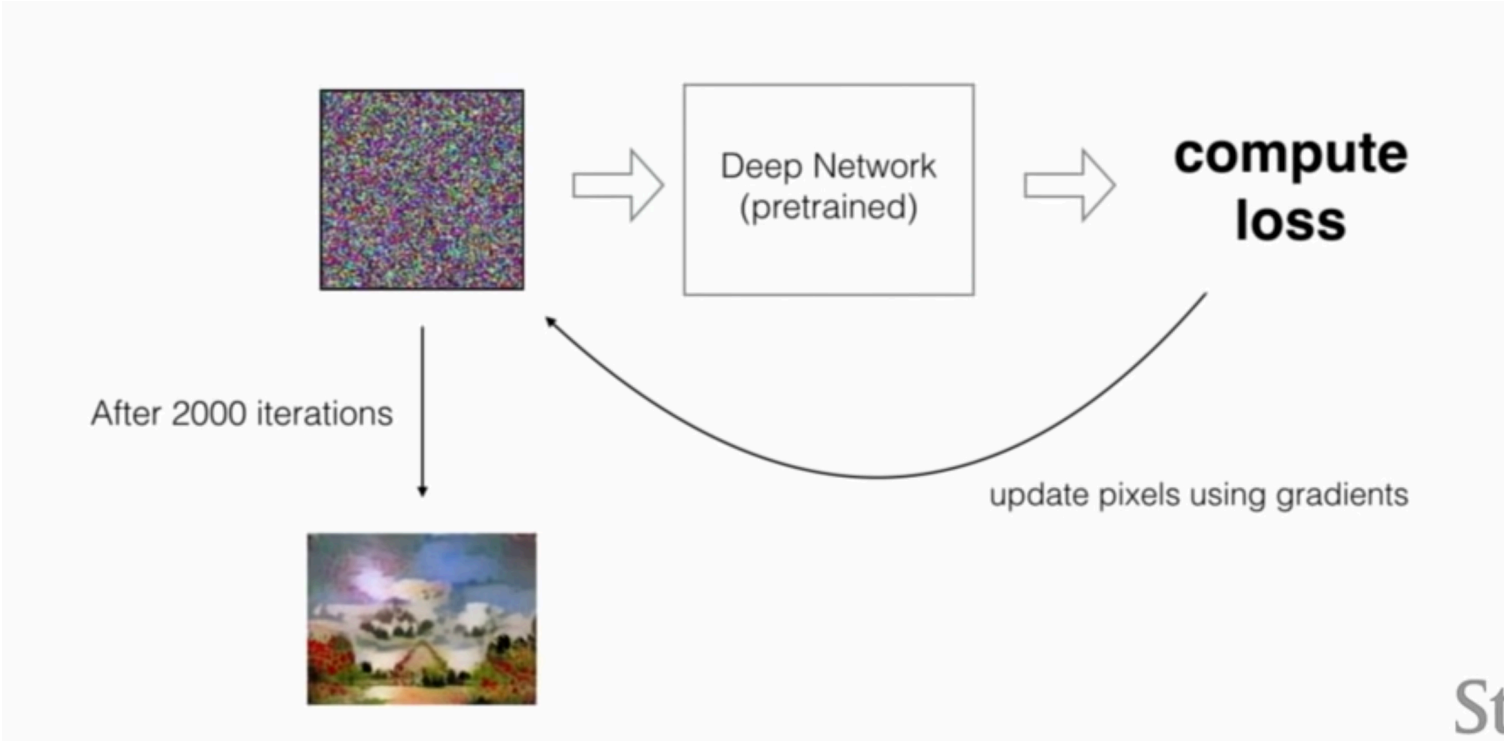
ContentC is content of content image
StyleC is style of style image
ContentG is content of generated image
StyleG is style of generated image

- **Loss**
  - We have three loss functions here.
  - We want the content of the content image look like the content of generated image
  - ➡ A = ContentC -ContentG => find a diffrent
  - ➡ B = StyleS - StyleG => find a diffrent
  - and minimize the differences of style of style image and style of generated image
  - ➡ Loss = A+B => ***Choose B***



4. Art Generation

- **Loss**



- We want to create new image
- => we not learning parameters by minimize L. Learning a new image
- Use ImageNet extract the contentC and styleS
- We use L to create (arrange) the pixel -> new image

5. Trigger word detection

- Given a 10sec audio, detect the word “activate”

- **Data**
  - A bunch of 10s audio clips
- **Input**

x = A 10sec audio clip

  - Green: positive word - “activate”
  - Pink: negative word
  - Yellow: left
- **Output**

y = 0 or y = 1

y = 00..0000**1**00000..000

y = 00..0000**1..1**000..000

  - Last activation ? -> **SIGMOID**

5. Trigger word detection

- **Data**

  - Data can be label:

  - And output is:

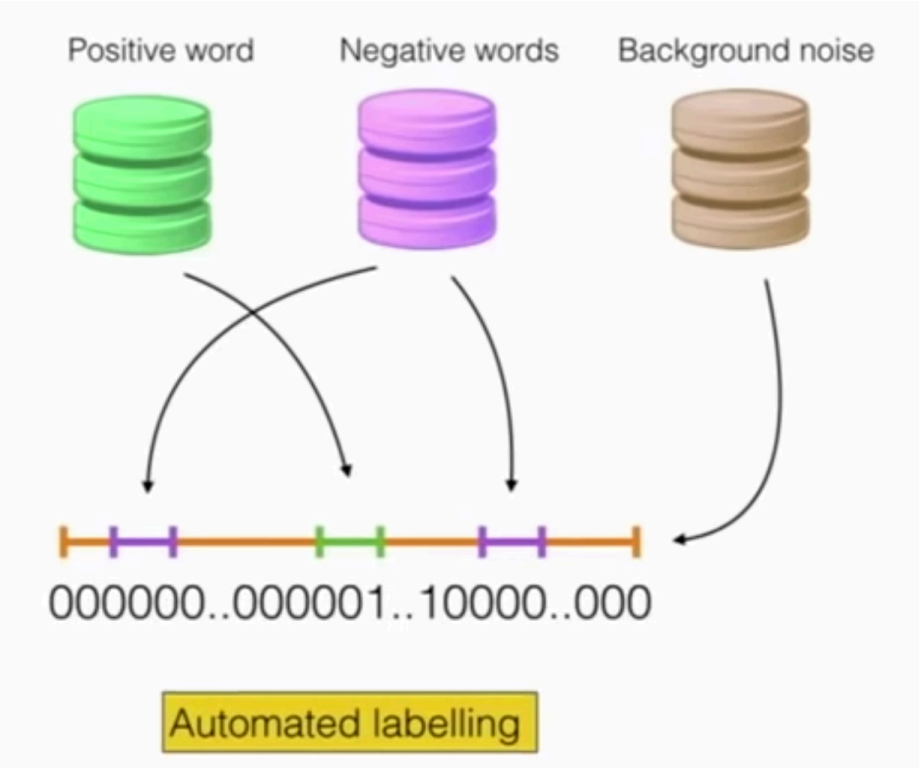
5. Trigger word detection

- **Architecture**
  - RCNN - Recurrent neural network
- **Loss**
  - L = -[y*log(y^) + (1-y)*log(1-y^)]

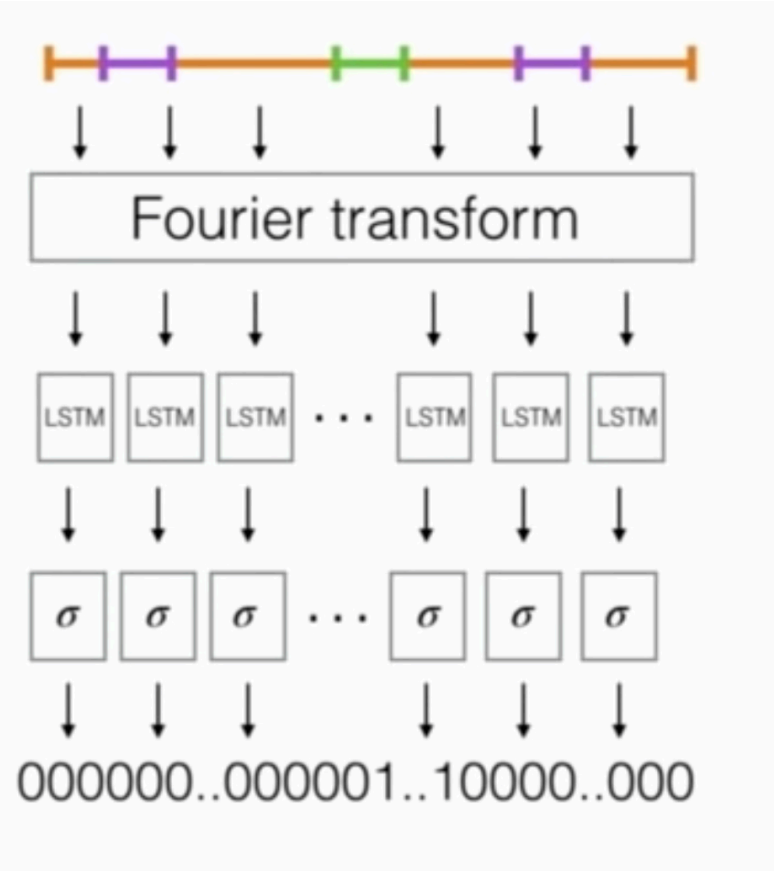
5. Trigger word detection

- **<u>Critical to success</u>**

  - Strategic data collection/ labelling process



  - Architecture search and Hyper parameter Tuning



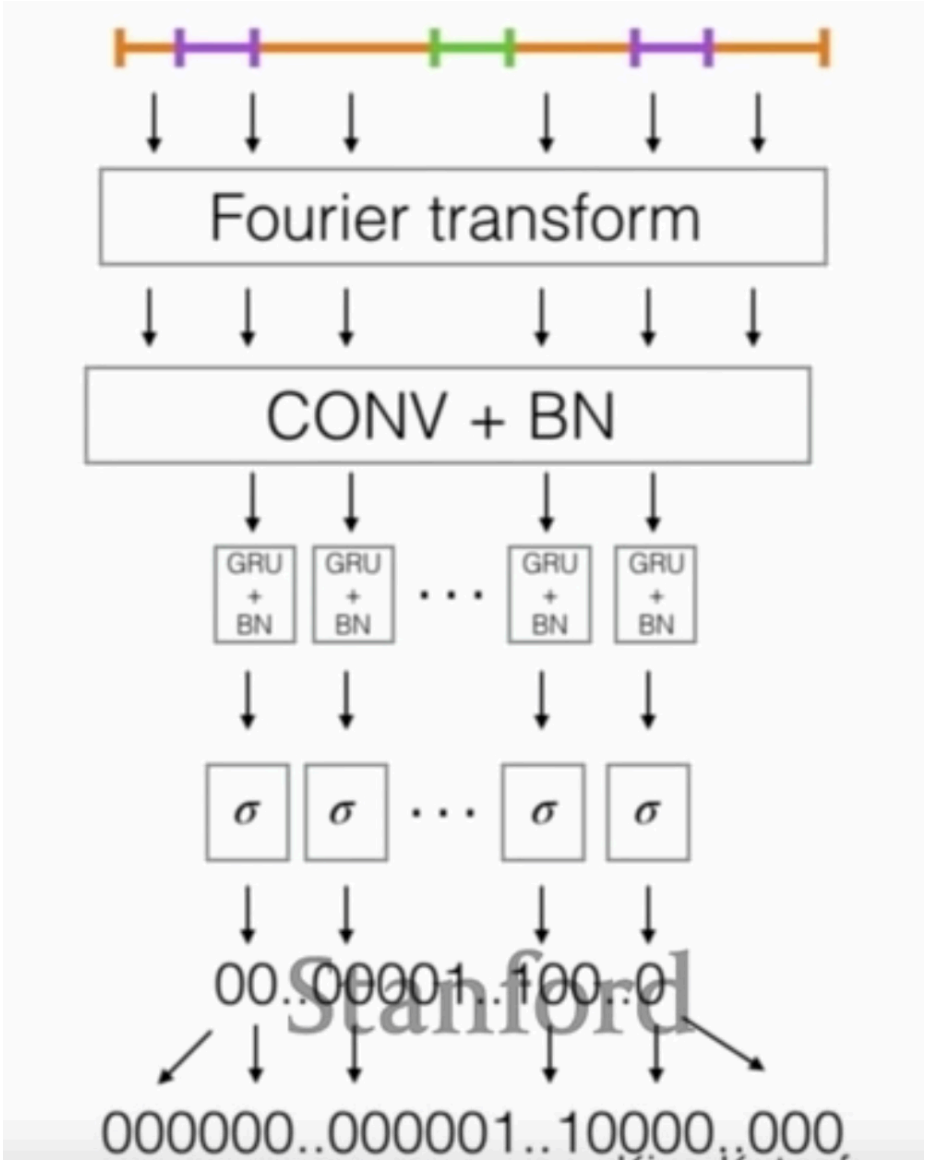
5. Trigger word detection

- **<u>Critical to success</u>**
  - Architecture search and Hyper parameter Tuning
    - **Issue:**

- Current neural network is too big => super hard to train use convolution to reduce a number of time step of audio clip
- Use batch Nor to specify type of layer
- Output time-steps is smaller than input, you must expand it

Go to this link to see a example

https://youtu.be/AwQHqWyHRpU?t=4702



6. Loss of bounding box - object detection

- **Architecture**

- Term 1: center of the bounding box x, y
- term 2: W, H stands for weight and height of bounding box => minimize true bounding box and predicted bounding box basically
- Term 3: has an indicator function with objects it say: ‘If there is an object, you should have a high probability for object ness”
- Term 4: no has object it say: ‘If there is no object, you should have a low probability for object ness”
- Final: term you find the class that is in this box, what class it belong

$$\lambda_{\text{coord}} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{\text{obj}} \left[(x_i - \hat{x}_i)^2 + (y_i - \hat{y}_i)^2 \right]$$

$$+ \lambda_{\text{coord}} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{\text{obj}} \left[\left(\sqrt{w_i} - \sqrt{\hat{w}_i} \right)^2 + \left(\sqrt{h_i} - \sqrt{\hat{h}_i} \right)^2 \right]$$

$$+ \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{\text{obj}} \left(C_i - \hat{C}_i \right)^2$$

$$+ \lambda_{\text{noobj}} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{\text{noobj}} \left(C_i - \hat{C}_i \right)^2$$

$$+ \sum_{i=0}^{S^2} \mathbb{1}_{i}^{\text{obj}} \sum_{c \in \text{classes}} \left(p_i(c) - \hat{p}_i(c) \right)^2$$

Stanford

6. Loss of bounding box - object detection

- **Architecture**

  - Why we use square root in weight, height ?
  - To penalize more the errors on small boxes than big boxes

$$\lambda_{\text{coord}} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{\text{obj}} \left[(x_i - \hat{x}_i)^2 + (y_i - \hat{y}_i)^2 \right]$$

$$+ \lambda_{\text{coord}} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{\text{obj}} \left[\left(\sqrt{w_i} - \sqrt{\hat{w}_i} \right)^2 + \left(\sqrt{h_i} - \sqrt{\hat{h}_i} \right)^2 \right]$$

$$+ \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{\text{obj}} \left(C_i - \hat{C}_i \right)^2$$

$$+ \lambda_{\text{noobj}} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{\text{noobj}} \left(C_i - \hat{C}_i \right)^2$$

$$+ \sum_{i=0}^{S^2} \mathbb{1}_{i}^{\text{obj}} \sum_{c \in \text{classes}} \left(p_i(c) - \hat{p}_i(c) \right)^2$$

Stanford

6. Reference

Stanford CS230: Deep Learning | Autumn 2018 |

Lecture 2 - Deep Learning Intuition

https://www.youtube.com/watch?v=AwQHqWyHRpU

4. Art Generation

- **Architecture**

◉ **Define the content image and style image:**

- The content image: we can detect the edges in the earlier layer, because the edges are usually a good representation of the content image, extract the information from first layer
- The style image: is non-localized information. It's a technique call Gram matrix