# AI6102 Direct Reading and Literature Review

**Liu Yaohong**
**Nanyang Technological University**
**G2203319C**
**liuy0220@e.ntu.edu.sg**

## 1 Background

Currently, I am working on a silicon package design similarity project. The objective of this project is to obtain the top view, front view, and side view images of two different types of silicon package designs and compare the similarity of the silicon chip stack-up designs using a deep learning model. Consequently, I selected this paper[9] related to image verification. This paper introduces a method for extracting features from images using a multi-scale CNN and curriculum learning, which helps improve the accuracy of image similarity detection. This report presents my reading process and summarizes the concept of "SimNet"[9].Some future work ideals is putting at the end of this report.

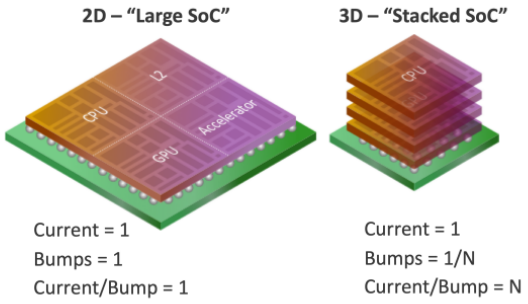


Figure 1: Silicon chip stack up design [7]

## 2 Introduction

In recent years, research, such as image classification, image verification has also been a hot topic. Different with image identification, the objective of image verification is determining the input pair of images whether they are similar or not.Image identification focus on image retrieval.The traditional image descriptors include **SIFT**[6], **LBP**[1],etc. In recent years, image embedding via **CNN** is a robust descriptor, which is able to learn those image features that are generated by itself.



### 2.1 Contributions

Prior research work was using traditional **VGG16**[8] as a feature descriptor. In this work, the concept of **SimNet** has been raised. It uses 2 different scales, shallow CNN, and together work with VGG16 in a Siamese network architecture. **SimNet** embeds pairs of images into 4069-dimensional subspace. Similar to **SVM** concept, it is trying to maximize the margin of negative match and minimize the distance of positive match. Therefore, choosing well trained image pairs is important for model training, which hugely affects the model accuracy. The traditional offline image pairs generation approach is inefficient. The author proposed an online pair mining strategy(**OPMS**). OPMS is able to increase the degree of training image pairs difficulty consistently. To summarize the contributions:
1. Brought up multi-scale CNN: CNN1 is VGG16. It is good at capturing high-level semantic features, like object categories, poses. By adding CNN2& CNN3, those shallower convulsion networks handle the-down-sampled images, which provide better color&shape features capturing.
2. The author introduced the 'online pair mining strategy': different from the traditional training method which is feeding the hardest negative pairs at the very beginning, the curriculum learning method started with the low level of difficulty of the negative pairs during the training, and feeding the hardest pairs at the end.it generating the image pairs in CPU on multi-threads, at same time training model at GPU. This method increase efficiency of resource usage and the model training speed.

## 3 Dataset

**CIFAR10**[5] is well labeled dataset. It contains 10 objects and each object has 6000 with size of 32x32 images. The dataset is splited into 5 training dataset and 1 test dataset evenly, which means each dataset contains 10000 color images. Especially for the test dataset, it randomly choose the 1000 image from each object. This work used 5-fold cross validation to tune hyper-parameters that is the purpose of split training dataset into 5. The test dataset used to validate the model at the end.

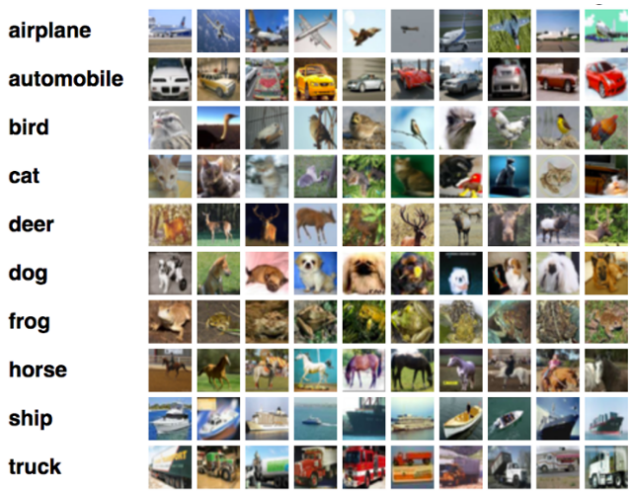


Figure 2: CIFAR10 Image Example

## 4 Methodology

This session we are going to discuss the model architecture.

### 4.1 Siamese Network Structure

Siamese Network is a pairing inputs network architecture. It use 2 CNN to embedding a pair of images. by minimizing the loss, to get the best parameters $\theta$ of function $x = f(I;\theta)$, which minimized the distance of those similar images. The network contains 1M to 150M parameters, it depends the CNN layer and size to be used. The **Siamese Network** architecture is shown on Figure3.

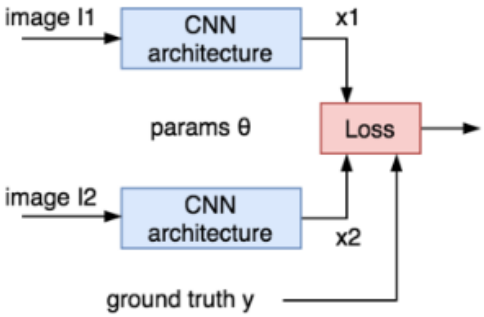


Figure 3: Siamese Network Architecture

To train **Siamese Network**,for image $I1\&I2$, a pair of image $(I_q, I_p)$ or a pair of negative image $(I_q, I_n)$ are fed to the network. The same object with 2 different views or different variations of the same type of image or applied data augmentation with the same object is considered as a pair of images. 2 images from different categories are a pair of negative images. To do so, the images $I_q, I_p, I_n$ have been projected on to position $x_q, x_p, x_n$. Hence, we can minimize the distance $(x_q, x_p)$ and maximize the distance $(x_q, x_n)$. Typical **Siamese Network** is using 2 CNN, each CNN has M layers and input image is d dimensions,$I \in \mathbf{R}^d$. after $m-1^{th}$ layer I has been mapped to $x^{m-1}$, on $M^{th}$ layer, $distance(x_q, x_p) = s(W^m x^{m-1} + b^m) \in \mathbf{R}^{pm}$.In space $\mathbf{R}^{pm}$, the similar images have small distance, and dissimilar images will be far way to each other. Details are shown in Figure4.

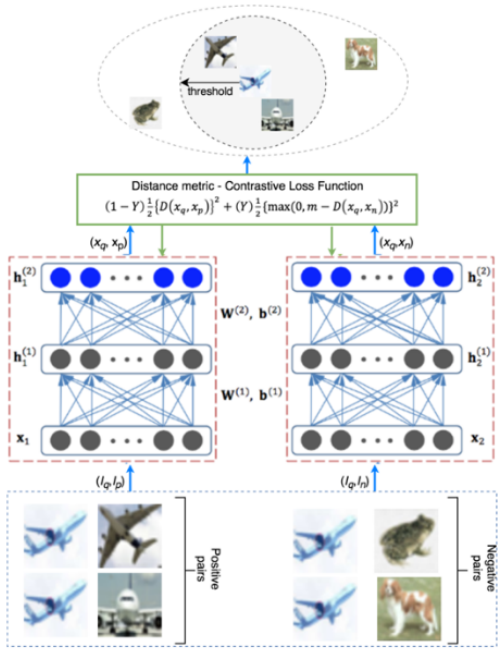


Figure 4: Siamese Network Architecture

**Contrastive Loss Function[4]** To optimize the network and measure the performance of $f$, the contrastive loss function has been applied. It measures how close those similar images and further apart the dissimilar images are. The equation of the contrastive loss function is Equation1:

$$L(\theta) = (1-Y)\frac{1}{2}\{D(x_q, x_p)\}^2 + (Y)\frac{1}{2}\{max(0, m - D(x_q, x_n))\}^2 \quad (1)$$

From Equation1, Y=0 when the input is similar images, $(Y)\frac{1}{2}\{max(0, m-D(x_q, x_n))\}^2$ term will become 0, so we can reduce the distance of similar images. Y=1 when input is dissimilar images. the term of $(1-Y)\frac{1}{2}\{D(x_q, x_p)\}^2$ will be omitted, left with $(Y)\frac{1}{2}\{max(0, m - D(x_q, x_n))\}^2$. when image is totally different, the loss equals 0, otherwise there is small loss. similar to SVM, m in Equation1 is the margin between similar and dissimilar images. Larger m makes similar and dissimilar images further. In this work, the author kept m=1.

### 4.2 Online Pairs Mining Strategy (OPMS)

Image pairing with label is important to Siamese Network training. The positive pair image $(I_q, I_p)$ needs to be labeled as 1 and the negative pair image $(I_q, I_p)$ is 0. For example, the dataset $I$ has $M$ categories, and each category gives $n$ images. In this case, the average number of positive pairs

and the number of negative pairs is $N_p = \sum_{i=1}^{M} \frac{n_i(n_i-1)}{2}$ and $N_n \sum_{i=1}^{M}(I-n)n$ respectively. there are 2 main problem using a pairs generated from I. firstly, With I increasing, $N_p$ and $N_n$ is multiple growth, which case model converge extremely slow. secondly, not all the pairs are useful to the model. In order to pick the "hard pairs image"5 to feed into training, the author introduced pair constraints and curriculum learning.

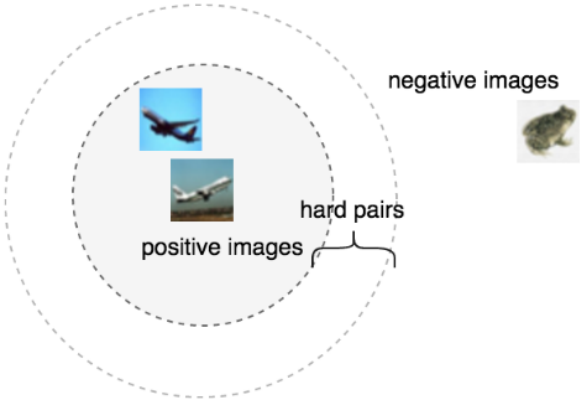


Figure 5: Image Hard Pairs

**Pair Constraint** To define the valuable image pairs, the pair constraint has been proposed. The equation of the pair constraint is shown on Equation2. Maximizing the distance between hard positive image $(I_q, I_p)$ and minimizing the distance between hard negative image $(I_q, I_n)$, which creates the boundaries between similar and dissimilar images. m is the margin between positive and negative pairs. If the images fall within the margin of below, by using pair constraint, they could be removed easily.

$$argmax\{D(x_q, x_p)\}^2 + m < argmin\{D(x_q, x_n)\}^2 \quad (2)$$

The Hyper-parameter $\lambda \in [0, 1]$ is used to control the difficult level of pairing of each mini batch which is fed into model training. By using this approach, the difficulty level can easily be increased. But this leads to a difficulty, which defines the difficult level of image pairing. On one hand, some pairs have 0 loss, which does not benefit the model at all; on the other hand, those poor images could be labeled as hard positive pairs or hard negative pairs. It will mislead the model training. To overcome this issue, the traditional method is offline generating image pair from the subsets of training dataset with every n step. This method requires you to save the model checkpoint at every n steps and put the pairs onto disk. To improve efficiency, the author proposed an online image generating approach. It generates image pairs from mini-batch and the generating is done on CPU. This approach increases CPU usage when GPU is using to train the model. The author selects all positive pairs in the mini-batch and in the negative pairs selection, the author gets the ideal form **curriculum learning**. Instead of feeding into the hardest negative pairs at the beginning of training,model is fed the hardest negative pairs at the end of training. This method makes models converge faster than the original method.

**Algorithm 1** Online Pair Mining Strategy OPMS

**Require:** *numEpochs* for network training

```
step = 1/numEpochs
λ = 0
for e do numEpochs
    λ+ = step
    for b do getImageMiniBatches()
        positivePairs = generatePositivePairs(b)
        negativePairs = generateRandomNegativePairs(b)
        curriculumSortedPairs = OPMS(positivePairs, negativePairs, λ)
        train model
    end for
end for
procedure OPMS(positivePairs, negativePairs, λ samplingWeight)
    margin m = 1
    calculate positivePairs and negativePair L2 euclidean distances
    sort pairs on L2 distance
    prune pairs argmax(D(x_q, x_p)^2) + m < argmin(D(x_q, x_n))^2 Eq. (2)
    negativePairs = weightedRandomSample(negativePairs, λ)
    pairs = CurriculumSortPairs(positivePairs, negativePairs)
    return pairs
end procedure
```

Figure 6: OPMS pesudocode

## Multi-Sacle CNN Structure

From Figure7, the author uses pretrained VGG16 to embed 224x224x3 image 4096D subspace. VGG16 is good at capturing high-level semantics features, like object categories, poses. To increase the model performance and the quality of image embedding, the author added 2 different scales of CNN to the original network. This is inspired by [11][2]. By adding CNN2(with output 512D)& CNN3(with output 1024D), those shallower convulsion networks handle the down-sampled images, which provide better color&shape feature capturing. To prevent 2 shallow CNN overfitting, L2 normalization has been applied as the following. Finally, 1024D & 4096D vector joint as 4096D vector.

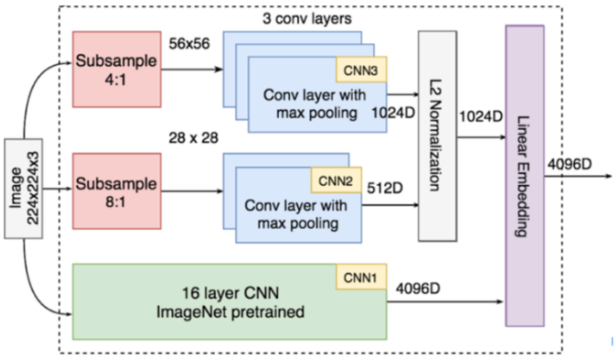


Figure 7: Multi-scale CNN architecture

## 5 Model Training

The author introduces the training and testing process of SimNet and the method to prevent overfitting is used for SimNet.

**Training** Author used VGG16(pretrained with imageNet data) in this experiment to make models converge faster. To control the updates easily, SGD optimizer has been chosen. It includes learning rate, weight decay and momentum helps model training with small steps, which helps to adjust the pretrained weights but will not wreck them. The author mentioned choosing the right number of epochs to prevent overfitting, which is the same ideal of early stop. Cross-validation is implemented to tune the hyper-parameters. Batch normalization is a good approach to shorten the model converge time.

**Overfitting** Image augmentation and dropout is applied to prevent model overfitting. Image augmentation with random transformation makes sure the same image is not fed into the model twice. Therefore, the correlation occurring in training data will be disrupted. Only the top 2 convolution layers were fine-tuned. Because fine-tuning all the layers leads to overfitting.

**Testing** in this experiment, hyper-parameters tuning was done by 5-fold cross validation. So there is no training trend to test set. In this case, test set can be used to test the final model generalized performance.

**Process** The trained model compares new images to a repository, ranking matches by L2 distance. It takes around 3 seconds per image on a GPU.

**Debugging** it is a good practice to plot layer weights, train loss, test loss per epoch and plot precision-recall curves, roc-auc curve for evaluation. By performing gradient checking, we will know whether back prorogation works well.

## 5 Experiment Setting& Results

By evaluating model on test set, the distribution of each category needs to be similar to the training data set. The author mentioned that the images are more complex, which conduct scales, occlusion, view points, deformation, intra-class variations, illumination etc problems in real life.

**Embedding space visualization** To clearly view the distances of similar images and dissimilar images, the author projects the final 4096 dimensional vector on to 2D space by using t-SNE8. From Figure**??**, we can see that SimNet generally performs well. The distances of different categories are clear. Those similar images are close to each other.

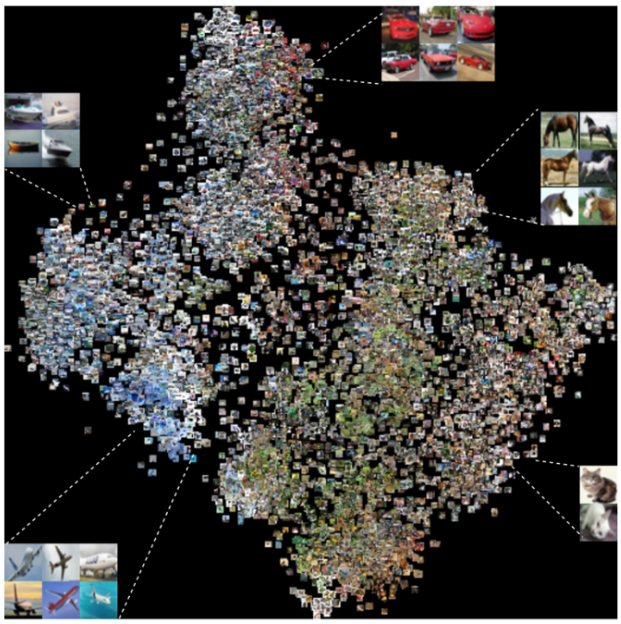

Figure 8: t-SNE 2D embedding space visualization of image

**Embedding dimensions** By trying out different dimensions(512D ,1024D ,2048D ,409DD) embedding, the author found that for shallower CNN, lower dimension embedding 512D and 1024D has better contribution for the SimNet. This cause by Vgg16 is used highest 4069D embedding to learn the high dimension information.

**Hyperparameter Settings** In this experiment, various image augmentations such as random crops, image rotation, zoom, Gaussian noise addition, affine transformations, contrast and brightness adjustments were applied randomly. SimNet was trained using the SGD optimizer with a learning rate of 1e-5, weight decay of 1e-6, and a Nesterov momentum of 0.9. To prevent overfitting, a dropout rate of 0.5 was applied to the fully connected layers. As mentioned earlier, the author fine-tuned only the top 2 layers of the pretrained Vgg16 model. Additionally, 2D zero padding of size (1,1) was applied to the image edges. The model was trained using a dataset ranging from 1.1 to 500 million pairs, and each training session consisted of 100 epochs. During this work, the author tried randomly selected pairs, adjusted the hyperparameter $\lambda$ within the range of [0.2,1], and feeding all image pairs methods. It was observed that the Euclidean distance metric outperformed cosine and L1 distance metrics during the training of SimNet.

**Evaluation** & **Performance** To compare SimNet with base line models, contrastive loss function with accuracy has been used in evaluation. Models‘ performance on test data set are shown in Table1.

| Model | Accuracy |
|---|---|
| A. Random guess in data | 10% |
| B. ImageNet pre-train classifier VGG16 baseline | 90.2% |
| C. Fine-tune; pre-train classifier VGG16 | 91.2% |
| D. ImageNet pre-train similarity VGG16 baseline | 90.28% |
| E. Siamese network; fine-tune pre-train VGG16 | 91.9% |
| F. Multi-scale Siamese fine-tune pre-train VGG16 with OPMS 5.5 million pairs | **92.6**% |

Table 1: Accuracy of each Models

From Table1, SimNet with OPMS shows the best performance. Compared with Siamess Net, the accuracy improved 0.7%.ImageNet pre-train classifier VGG16, the accuracy of SimNet with OPMS improved 2.4%

Instead of feeding all the possible image pairs, OMPS is trying to make model learn the most valuable pairs so that we can reduce the training time. As discussed previously, OMPS learning is inspired by curriculum learning. By controlling $\lambda$, the model will learn the hardest pairs at the end of the training process. With this policy, model can learn better and efficiently. The result of tuning $\lambda$ is shown on Table2.

| Pair formation method | # of pairs (million) | Accuracy | Train (days) |
|---|---|---|---|
| A. OPMS with $\lambda$=0.2 | 1.1 | 81.0% | 1 |
| **B. OPMS with $\lambda$=0.5** | **5.5** | **92.6**% | **3** |
| C. OPMS with $\lambda$=0.7 | 10.1 | 90.2% | 6 |
| D. OPMS with $\lambda$=0.9 | 15.7 | 90.4% | 7 |
| E. OPMS with $\lambda$=1.0 | 21.3 | 92.8% | 12 |
| F. Randomly pairs | 5.5 | 78.1% | 3 |
| G. All pairs | 500 | - | ≈378 |

Table 2: Impact of Training Time

From Table2, when $\lambda = 0.2$ , the training dataset cannot provide enough pairs, which causes the model underfitting, the accuracy is only 81%. When $\lambda = 0.7\&0.9$, the accuracy has only less increasing, the model has stagnated. With $\lambda = 1.0$, model trained with 21.3 million pairs image, the model reached the best performance; accuracy equals 92.8%. But the whole training process takes 12 days. When $\lambda = 0.5$, 5.5 million pairs are used to train the model, the accuracy reaches 92.6%. Compared with the model $\lambda = 0.1$, the accuracy drops 0.2&, but the days of training decrease from 12 days to 3 days, which makes the model converge 4 times faster. The author has tested randomly selected 5.5 million pairs as comparison. The result is shown on model F with accuracy 78.1%. This shows that OPMS with controlling $\lambda$ significantly affects the model performance. In the end, the experiment did not test feeding all image pairs, which is more than 500 million pairs of image. But the author has written the approximated train days. It is around 378 days, which is a large number and this is not sustainable.

## 6. Future Work

From previous discussion we know, the challenge of image similarity is training time and the accuracy. With training CIFAR10[5], the model takes 3 days to maintain accuracy. But in the real world, there are more noisier and complex images. To further improve the model performance and reduce training time, instead of using Vgg16, the network with less parameters like ResNet[3] or GoogLeNet[10] has been introduced by author. There are 2 main reasons:

1. Due to the usage of global average pooling rather than fully-connected layers, ResNet has 20x parameters lesser than Vgg16 which make training time even faster.
2. Due to spacial residual block and skip connection architecture, ResNet can go deeper than Vgg16. ResNet has the better performance not only in capturing the high dimensions information but also the information on lower dimensions.

## 7. Conclusion

In this work, the author have introduced SimNet as a solution for determining image similarity based on a given reference image. SimNet demonstrated better capability in image verification work than CNN.The 'Online Pair Mining Strategy' helps model convergence to achieve 4x faster than original training time with scarifies minimum model performance .

## References


[1] Ahonen, A. H., Timo; and Pietikainen, M. 2006. Face description with local binary patterns: Application to face recognition. 2037–2041.

[2] C. Farabet, L. N., C. Couprie; and LeCun, Y. 2013. Learning hierarchical features for scene labeling. 1915–1929.

[3] et. al, K. H. 2015. ResNet – deep residual learning for image recognition.

[4] Hadsell, C. S., R.; and Lecun, Y. 2006. Dimensionality Reduction by Learning an Invariant Mapping.

[5] Krizhevsky, A.; and Hinton, G. 2009. Learning multiple layers of features from tiny images.

[6] Lowe, D. G. 2004. Distinctive image features from scale-invariant keypoints. 91–110.

[7] Saurabh Sinha, M. B. S. D. B. C., Xiaoqing Xu; and Yeric, G. 2020. Stack up your chips: Betting on 3D integration to augment Moores Law scaling. 10866.

[8] Simonyan, K.; and Zisserman, A. 2014. Very deep convolutional networks for large-scale image recognition.

[9] Srikar Appalaraju, V. C. 2017. Image similarity using Deep CNN and Curriculum Learning.

[10] szegedy et. al, C. 2014. GoogLeNet – going deeper with convolutions.

[11] Wang, e. a., Jiang. 2014. Learning fine-grained image similarity with deep ranking.