DeViSE: A Deep Visual-Semantic Embedding Model by ML-Team.47

Wo-Chien Lin [*1] **Chen-Chia Huang** [*2]

Abstract

Recently, Many visual recognition model are published, and many of them can achieve high accuracy in recognizing objects that are the classes among training labels. However, while the training labels have only 1000 classes, the objects in our real life have tens of millions of classes. Thus, our model aims to achieve zero-shot which is able to recognize the objects out of training labels with the combination of visual recognition model and word embedding. Although our idea and our goals are inspired from DeVise, we have adopted a different method from it, which is in some case simpler than method of theirs, and our method indeed performs well in zero-shot learning task.

1. Introduction

Since the visual world is consisted of a large number of objects, a common or a simple visual recognition method is not strong enough to recognize all of them.The main reason is that those common visual models can only classify n discrete label in the training data set, which makes the performance limited. Against this problem, some researchers further come up with some methods that can recognize the objects which is not the label among training dataset, called zero-shot.

Among zero-shot, some people stated a idea which considers the relationship between all the visual objects. Such as DeVise, it uses the visual model to recognize the image then feeds the result into semantic word embedding. Motivated by this idea, a modified model called ConSE is proposed, which is easier for people to build it and according the paper, it contributes a better accuracy in zero-shot. Thus, we aim to achieve zero-shot based on this model, and make some additional modification to try to improve accuracy.

[*]Equal contribution [1]Google, New London, Michigan, USA [2]National Tsing Hua University, Taiwan, Hsinchu. Correspondence to: Wo-Chien Lin <j890111tw@googol.com>, Chen-Chia Huang <lbnair366@gmail.com>.

Proceedings of the 34^{th} International Conference on Machine Learning, Hsinchu, Taiwan, PMLR 70, 2017. welcome Copy 2017 by the author(s).

2. Baseline model

2.1. Dataset link

- *Visual model*
  Cifar-100 http://www.cs.utoronto.ca/ kriz/cifar.html

- *Word Embedding*
  Wiki 2014 https://nlp.stanford.edu/projects/glove/

2.2. How to import data

The image data we downloaded from cifar-100 is end with ".pkl" , so we use function **unpickle** to turn it into a linear matrix. Next, we use the function called **get-cifar100** to turn the matrix into $32 * 32$ with 3 channels(RGB). Finally , we use **load-cifar100-data** to resize the image into $224 * 224$ with 3 channels(RGB) for the reason that the ResNet could only read the image size with $224 * 224$.The resizing process is illustrated in Figure 1.

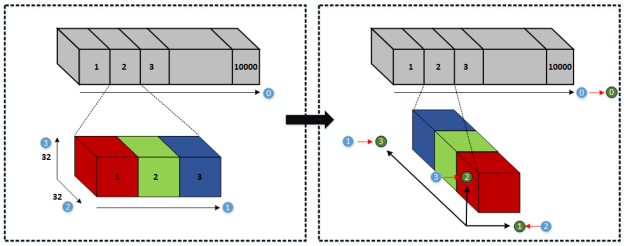

Figure 1. Resize of image

Additional library
–pickle
–Gensim
–keras
–sklearn
–matplotlib
–PIL
–glob

–opencv

-Other environment setting
–python3.5
–Tensorflow 1.4 GPU version
–notebook 4.23
–CUDA 8.0
–CUDNN

2.3. Network structure

Our goal is to develop a model which can learn few classes from training image , and after training , when we feed a unseen image , it can still tell us what is in the image.

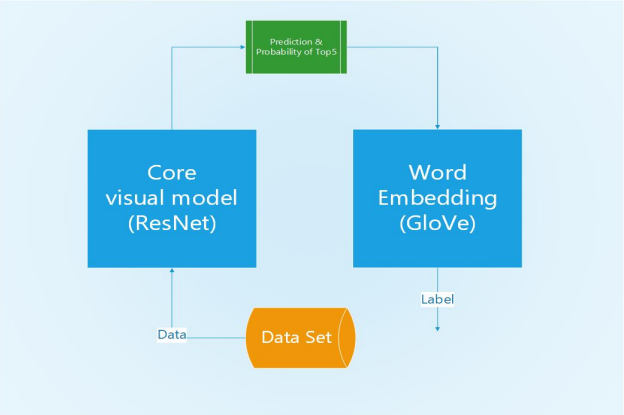

Figure 2. Network structure

As shown in *Figure 2.* , our model is consisted of two parts:

1. Core visual model
   It is constructed by using pre-trained ResNet-50 model , and we then trained it with cifar-100 which contains 100 classes with 500 training images and 100 testing images for each class. However , we only use 3000 training data and 1000 test data in cifar-100 because if we use too many images in the cifar-100 , the model will overfit to cifar-100 , which implies that the original pre-trained model would totally get loss. Figure 3. shows one of the result of this core visual model. Figure 4. shows the accuracy of our baseline visual model.

2. Word Embedding
   The word embedding model is constructed with GloVe's pre-trained 300 dimension model, and it is imported by using KeyedVectors , which is from Gensim library. Figure 5. shows one of the result of this word embedding.

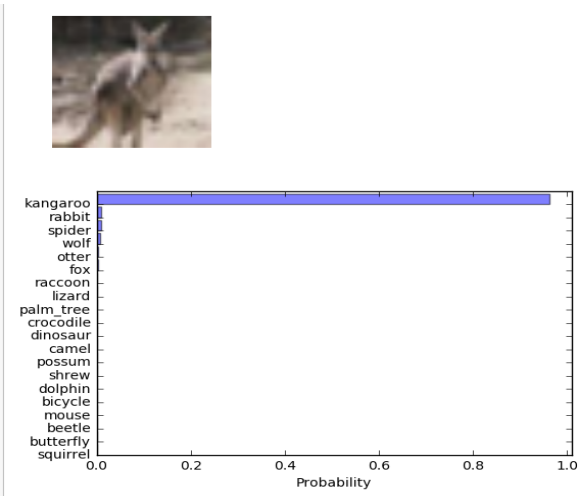

Figure 3. Result of visual model

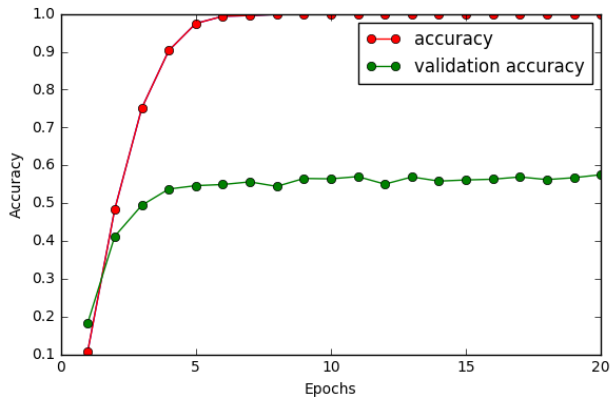

Figure 4. Training/Validation accuracy per epoch

2.4. Result

The table below refers to flat hit rate of top k. That is, if there is the true label among the prediction of top k, we regard it as correct recognition, so along with k bigger, the accuracy might getting higher too.

Flat hit@k (%)	k=1	k=2	k=5	k=10	k=20
Accuracy	0%	2.5%	10%	17%	23.5%

Figure 6. shows the result of our baseline model. In general, the visual model can roughly recognize what the object is in the image. However, for the picture of the car, although our model can predict the brand of it called suv, however, we think it might just be a coincidence, and for the mountain and the boy, although the prediction is correct ,it is not zero-shot because the label is among the classes of

```
]: y2 = model.most_similar("apple", topn=20)
   for item in y2:
       print (item[0], item[1])
```

```
iphone 0.5987042188644409
macintosh 0.5836331248283386
ipod 0.5761123895645142
microsoft 0.5663833022117615
ipad 0.5628098249435425
intel 0.5457563400268555
ibm 0.5286195278167725
google 0.5282472372055054
imac 0.5072520971298218
software 0.4962984323501587
motorola 0.47161784768104553
computer 0.4711154103279114
apples 0.46574175357818604
itunes 0.4646926522254944
pc 0.4600933790206909
iphones 0.452551007270813
mac 0.4503524601459503
ipods 0.44740575551986694
cherry 0.4466744031429291
computers 0.44292744994163513
```

Figure 5. Result of Word Embedding

cifar-100. As the result, our baseline model is not good enough for zero-shot task, so we further purposed to make progress on it. The improved model will be introduced in the next section.

Figure 6. Result of baseline model

3. Improved Model

Our improved model is referred to the method proposed in ”Zero-shot learning by convex combination of semantic embeddings”, called ConSE, which is reported better than the performance of DeViSe. Simultaneously, we also improved the accuracy of our core visual recognition model, which performs the accuracy of 65 percent(as Figure 7.) higher than the that of original model, which is 53 percent. By the way, we makes no change to the dataset of both visual model and word embedding.

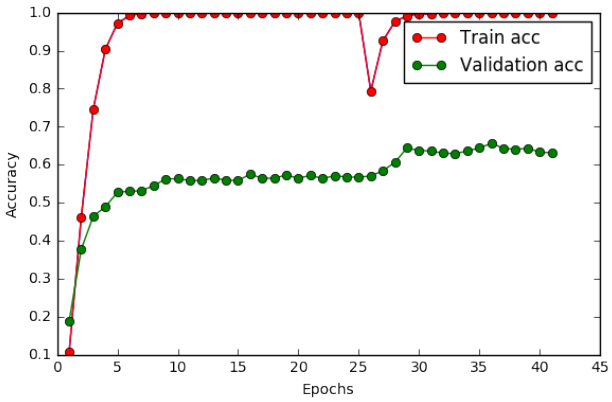

Figure 7. Improved training/validation accuracy per epoch

3.1. Network structure

As shown in Figure 8. our improved model can separated into three parts:
-Core visual model
-Word Embedding
-Multiplex(Probability, X)

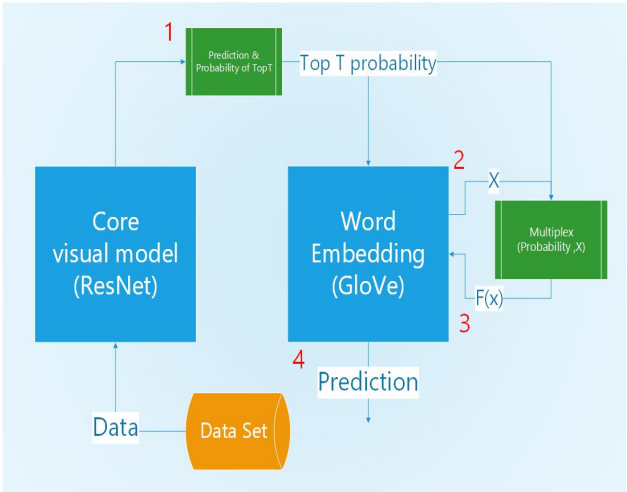

Figure 8. Structure of improved model

Both core visual model and word embedding is mostly same as those of baseline model, except for the improved accuracy of visual model. The main reason that our model is advanced is because of *Multiplex(Probability, X)*. Next we will introduce our improved model step by step with formula.

1. **Prediction & Probability of Top T**
   After softmax layer of visual model, there would be several prediction, and for the one that have the highest probability, called $\hat{y_0}(x, 1)$. Formally, we denote

$$\hat{y_0}(\mathbf{x}, 1) = \underset{y \in \mathsf{y}_0}{\operatorname{argmax}} p_0(y|\mathbf{x}). \tag{1}$$

   Similarly, let $\hat{y_0}(x, t)$ denote the t^{th} most likely training label for **x** according to p_0.In other words, $p_0(\hat{y_0}(x, t)|x)$ is the t^{th}largest value among $\{p_0(y|x); y \in \mathsf{y}_0\}$. Additionally, according to the research, top 10 could receive better accuracy, so in this project, we adopt top 10 in our model.

2. **Top T to Word Embedding**
   In this part, we feed the top T prediction respectively into word embedding model. As we can imagine, there will be T sets of embedding vectors. For convenience, we called it X, and X will feed into the function *Multiplex(Probability, X)*.

3. **Multiplex(Probability, X)**
   After feeding top T into word embedding, we have both probability and embedding vectors for each prediction. That is, we have T sets of probability and embedding vectors. For each set, we multiplex the embedding vectors and corresponding probability as their weight. More formally,

$$f(\mathbf{x}) = \frac{1}{Z} \sum_{t=1}^{T} p(\hat{y_0}(\mathbf{x}, t)|\mathbf{x}) \cdot s(\hat{y_0}(\mathbf{x}, t)) \tag{2}$$

   where $s(\hat{y_0}(\mathbf{x}, t))$ denote the semantic embeddings of $\hat{y_0}(\mathbf{x}, t)$, and Z is a normalization factor given by

$$Z = \sum_{t=1}^{T} p(\hat{y_0}(\mathbf{x}, t)|\mathbf{x}) \tag{3}$$

   If the classier is very condent in its prediction of a label y for **x**, i.e., $p_0(y|x) \approx 1$, then $f(x) \approx s(y)$. However, if the classier have doubts whether an image contains a lion or a tiger, e.g., $p_0(lion|\mathbf{x}) = 0.6$ and $p_0(tiger|\mathbf{x}) = 0.4$, then our predicted semantic embedding, $f(x) = 0.6 \cdot s(lion) + 0.4 \cdot s(tiger)$, will be something between lion and tiger in the semantic space. Even though liger (a hybrid cross between a lion and a tiger) might not be among the training labels, because it is likely that $s(liger) \approx \frac{1}{2}s(lion) + \frac{1}{2}s(tiger)$, then it is likely that $f(x) \approx s(liger)$.

4. **Output prediction**
   After the function of *Multiplex(Probability, X)*, we put f(**x**) into word embedding again. In this step, we perform zero-shot classification by finding the class labels with embeddings nearest to f(**x**) in the semantic space. The top prediction of our model for an image **x** from the test label set, denoted $\hat{y_1}(x, 1)$, is given by

$$\hat{y_1}(\mathbf{x}, 1) = \underset{y' \in \mathsf{y}_1}{\operatorname{argmax}} cos(f(\mathbf{x}), s(y')) \tag{4}$$

   We use cosine similarity to rank the embedding vectors. Moreover, let $\hat{y_1}(\mathbf{x}, k)$ denote the k^{th} most likely test label predicted for **x**.

   Similarly, previous work on zero-shot learning also uses a similar k-nearest neighbor procedure in the semantic space to perform label extrapolation. The key difference in our work is that we define the embedding prediction f(**x**) based on a standard classifier as in Eq. (2), and not based on a learned regression model.

3.2. Table for Result

The table below shows the comparison of flat hit rate between baseline and improved model, for k= 1, 2, 5, 10, 20. As we can see, improved model is no doubt performs much better than baseline model.

Flat hit@k (%)	k=1	k=2	k=5	k=10	k=20
Baseline	0%	2.5%	10%	17%	23.5%
Improved	2.5%	5%	12.5%	27.5%	32.5%

4. Result

After introducing our improved model, here we shows some results of it, as Figure 9. These images are respectively tomato, taxi and piano, and indeed, they are the objects out of classes of cifar-100. It is obvious that the improved model can recognize those objects, for that they are predicted correctly within top 20 prediction. Moreover, we did a small modification which is removed the prediction that is one of cifar-100 classes. By do so, since there may be three or even four predictions removed, the rank can move forward, which further increase the probability of hitting the true label. One of the successful example, as shown in figure 9., is the piano which is exactly hit at the rank of k=20.

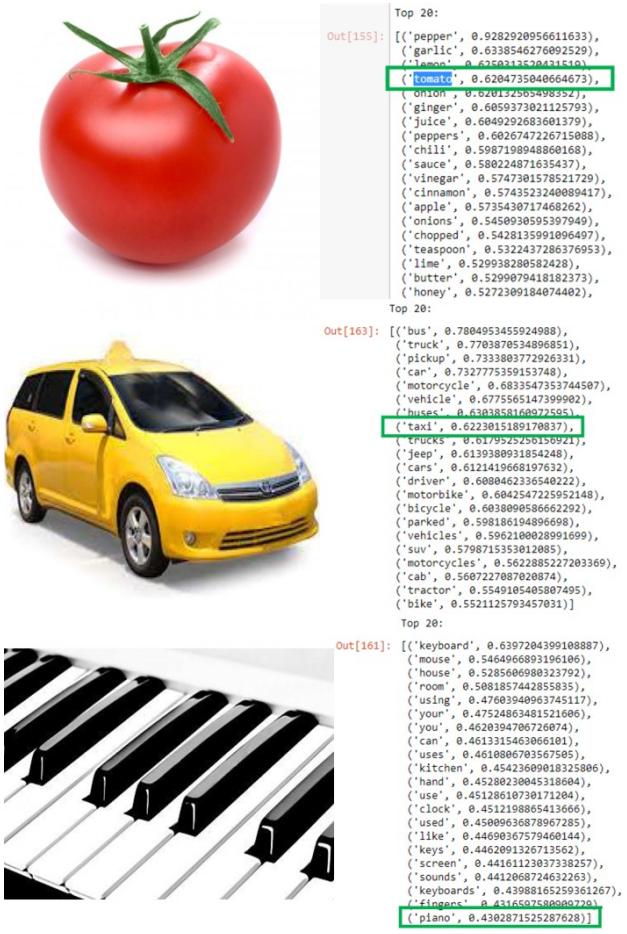

Figure 9. Result of improved model

Note that baseline model is just putting the prediction with highest probability from the visual model into word embedding, and output the result. Compared to baseline model, we can imagine that the improved model takes reference to top 10 predictions of visual model, consider characteristics of those predictions, and finally make a guess for the image. Surprisingly, this recognition process is similar to human, who would combine currently known knowledge and then make a guess.

5. Conclusion

As we stated earlier, the improved model does really outperform baseline model. However, the performance of improved model is still much lower than DeVise. We think that the main reason is that our dataset is cifar-100 with only 100 classes lower than that of ImageNet which contains 1000 classes, so the accuracy is limited.

As the research of "Zero-shot learning by convex combination of semantic embeddings", this approach can performs better than DeVise. However, since the dataset we use is different from the dataset DeVise used, we cannot compare our model with it. As the result, we proposed to change our dataset to ImageNet in the future, and further compare to DeVise, in order to make a higher accuracy than DeVise did.

References

A. Farhadi, I. Endres, D. Hoiem and Forsyth, D. Describing objects by their attributes. Technical report, 2009.

A. Krizhevsky, I. Sutskever and Hinton, G. Imagenet classication with deep convolutional neural networks. Technical report, 2012.

Bart, E. and Ullman, S. Cross-generalization: learning novel classes from a single example by feature replacement. Technical report, 2005.

C. H. Lampert, H. Nickisch and Harmeling, S. Learning to detect unseen object classes by between-class attribute transfer. Technical report, 2009.

G. Corrado, J. Shlens, S. Bengio J. Dean M. Ranzato T. Mikolov. Devise: A deep visual-semantic embedding model. Technical report, 2013.

J. Deng, W. Dong, R. Socher L.-J. Li K. Li and Fei-Fei, L. Imagenet: A large-scale hierarchical image database. Technical report, 2009.

Jason Weston, Samy Bengio and Usunier, Nicolas. Large scale image annotation: learning to rank with joint word-image embeddings. Technical report, 2010.

M. Norouzi, T. Mikolov, S. Bengio Y. Singer J. Shlens. Zero-shot learning by convex combination of semantic embeddings. Technical report, 2013.

R. Socher, M. Ganjoo, H. Sridhar O. Bastani C. D. Manning and Ng, A. Y. Zero-shot learning through cross-modal transfer. Technical report, 2013.

T. Mensink, J. Verbeek, F. Perronnin and Csurka, G. Metric learning for large scale image classication: Generalizing to new classes at near-zero cost. Technical report, 2012.