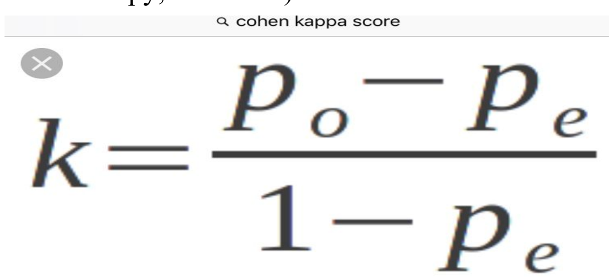
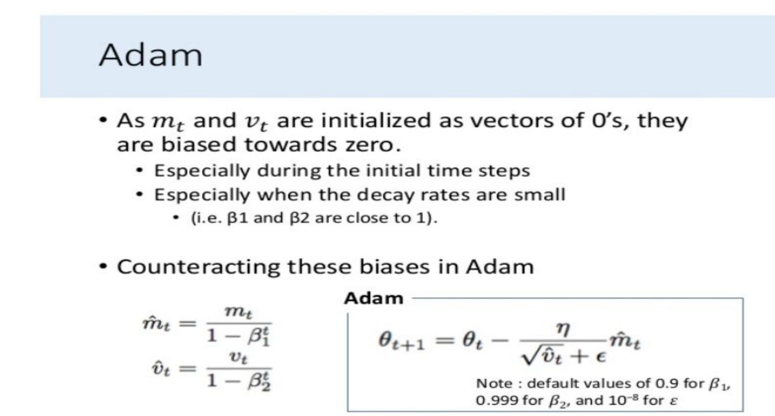
Honey Bee Image Classification

DATS6203 Machine Learning 2, Prof: Amir Jafari
Group 4: Ruyue Zhang

This project goal is to use deep learning network to classify Bees Subspecies and Hive Health. The data is from Kaggle: https://www.kaggle.com/jenny18/honey-bee-annotated-images/kernels
In this project, we first explore the dataset and operate data preprocessing. Then we use 2 different neural network frameworks to classify Bee Subspecies and Hive Health. In this project, we use Gradient Descent Algorithm to optimize the network. In the end we evaluate the result by f1 score, AUC and ROC and test accuracy. Yijia did the load image function and worked with Phoebe on Pytorch CNN model for health and subspecies. Xiaochi did the Keras cnn model for health. Phoebe and Xiaochi together finished EDA. I finished the Keras cnn model for subspecies first testing most models with hyperparameters then offering advice for health Keras model and Pytorch model.In terms of portion, I suppose is ¼.
Basically after preprocessing the data, I experimented with building cnn models for subspecies with different hyperparameters .
I started a basic convolutional neural(with 2 conv total 8 layers) network then adding layers, changing kernel size, trying different optimizers, learning rate, loss function, weight initialization, and activation. I tested around 19 models.
The initial training algorithm I used is Adam. It belongs to gradient descent but with adaptive learning rate. I tried sgd as well but not the standard sgd. Transfer function tried: softmax, relu, sigmoid. Loss function tried: binary_cross_entropy, categorical_cross_entropy, two custom loss function (weighted cross entropy, focal loss). Metrics used to measure accuracy: F1 score, Cohen kappa score .





$$F_1 = 2 \cdot \frac{\text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}}$$

$$H_p(q) = -\frac{1}{N} \sum_{i=1}^{N} y_i \cdot log(p(y_i)) + (1 - y_i) \cdot log(1 - p(y_i))$$

Binary Cross-Entropy / Log Loss

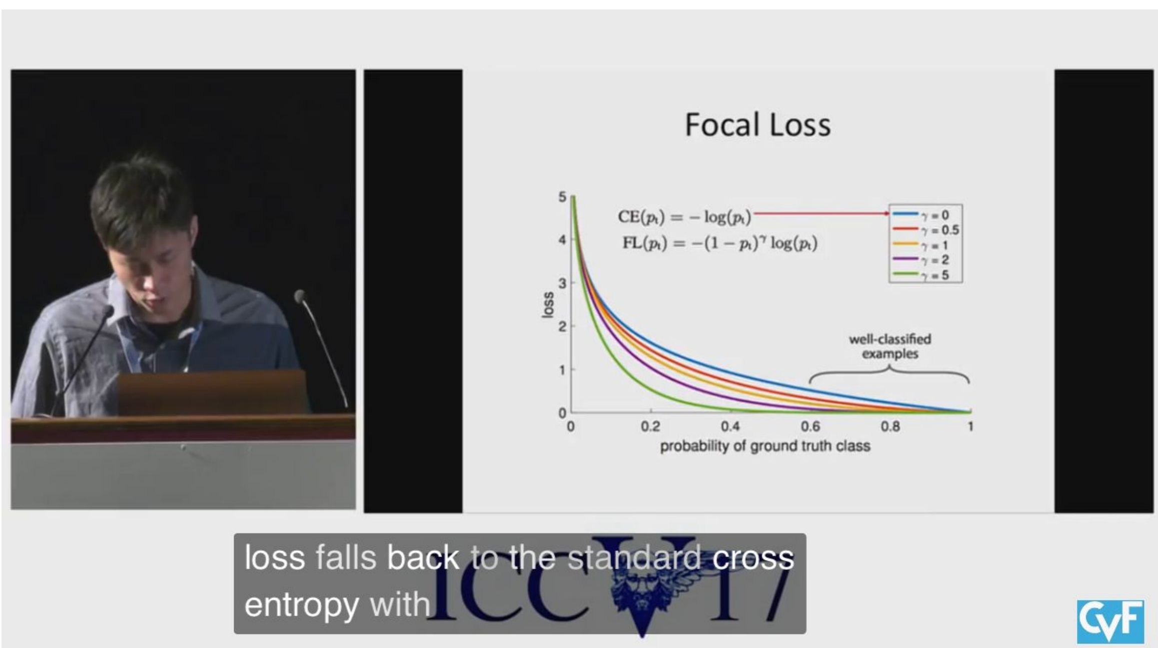
$$L_i = -\log\left(\frac{e^{f_{y_i}}}{\sum_j e^{f_j}}\right)$$

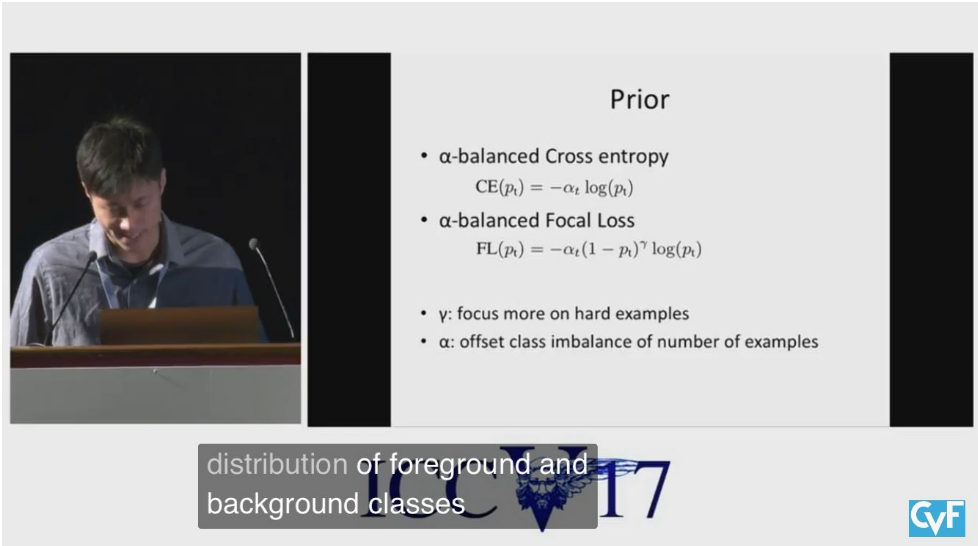
$$H(p, q) = -\sum_{x} p(x) \log q(x)$$

SuperDataScience

Convolutional Neural Networks (CNN): Softmax & Cross-Entropy

Focal loss





Workflow:

Basic cnn model ---- trying different epochs (20,30,50)---- trying activation (softmax,relu,sigmoid) ---- trying optimizers(adam, sgd) ----trying adding sample weights ---- trying adding convolutional layer (2-4) ----trying kernel size (5,3) ----trying different kernel initialization (glorot uniform,he_normal) ----trying adding other layers(dropout, batch normalization) ---- trying custom loss function for imbalanced data(weighted CE loss, Focal loss) ---- pick the best model changing the learning rate and batch size

Experiment data:

model_name	Layers	ACC	F1	Kappa	Kernel size	Kernel initialization	Optimizer	Loss fuc	Activation	Time (s)	Epoch	Batch size
modelS	2 conv total 8	0.90	0.91	0.86	5,5,3,3	Glorot uniform	Adam	Categorical_CE	softmax	166	50	32
modelS	2 conv total 8	0.89	0.90	0.83	5,5,3,3	Glorot uniform	Adam	Categorical_CE	softmax	67	20	32
modelS	2 conv total 8	0.89	0.89	0.824	5,5,3,3	Glorot uniform	Adam	Categorical_CE	softmax	95	30	32
modelR	2 conv total 8	0.56	0.41	0.0	5,5,3,3	Glorot uniform	Adam	Categorical_CE	Relu	177	50	32

model_sg	2 conv total 8	0.97	0.91	0.86	5,5,3,3	Glorot uniform	Adam	binary_CE	sigmoid	179	50	32
modelS_S	2 conv total 8	0.77	0.72	0.059	5,5,3,3	Glorot uniform	SGD	Categorical_CE	softmax	165	50	32
modelS_sa	2 conv total 8	0.91	0.91	0.86	5,5,3,3	Glorot uniform	Adam	Categorical_CE	softmax	164	50	32
model45	4 conv total 15	0.91	0.91	0.86	5,5,3,3,3,3, 2,2,	Glorot uniform	Adam	Categorical_CE	softmax	134	37	32
model43	4 conv total 15	0.92	0.93	0.88	3,3,3, 3,3,3, 2,2	Glorot uniform	Adam	Categorical_CE	softmax	178	50	32
model43_h	4 conv total 15	0.916	0.92	0.867	3,3,3, 3,3,3, 2,2,	he_normal	Adam	Categorical_CE	softmax	83	27	32
model43_d	4 conv total 15	0.918	0.92	0.873	3,3,3, 3,3,3, 2,2,	Glorot uniform	Adam	Categorical_CE	softmax		33	32
model33_b	3 conv total 18	0.88	0.89	0.82	3,3,3, 3,2,2	Glorot uniform	Adam	Categorical_CE	softmax	187	30	32
model43_b	3 conv total 23	0.90	0.91	0.859	3,3,3, 3,3,3, 2,2	Glorot uniform	Adam	Categorical_CE	softmax	263	47	232
model43_bdr	4 conv total 23	0.92	0.93	0.88	3,3,3, 3,3,3 2,2	Glorot uniform	Adam	Categorical_CE	softmax	314	48	32
model43_w	4 conv total 15	0.89	0.89	0.82	3,3,3, 3,3,3, 2,2	Glorot uniform	Adam	weighted_categorical_CE	softmax	83	22	32
model43_f	4 conv total 15	0.93	0.93	0.89	3,3,3, 3,3,3 2,2	Glorot uniform	Adam	Focal loss	sigmoid	180	50	32
model431r005	4 conv total 15	0.86	0.86	0.77	3,3,3, 3,3,3, 2,2,	Glorot uniform	Adam	Categorical_CE	softmax	185	50	32

model43lr0007	4 conv total 15	0.92	0.93	0.88	3,3,3, 3,3,3, 2,2,	Glorot uniform	Adam	Categorical_CE	softmax	178	50	32
model43lrchaing	4 conv total 15	0.74	0.70	0.54	3,3,3, 3,3,3, 2,2,	Glorot uniform	SGD	Categorical_CE	softmax	171	50	32
tmodel4364	4 conv total 15	0.91	0.92	0.86	3,3,3, 3,3,3, 2,2	Glorot uniform	Adam	Categorical_CE	softmax	164	48	64

In terms of transfer function, or this dataset, according to f1,softmax>sigmoid>relu. For batch_size, 32 performs better than 64. For Kernel initialization, the default Glorot uniform is better than he_normal. Initial kernel size 3 is better than 5. Adding more convolutional layers reaches higher f1 score. Adam performs way better than SGD.

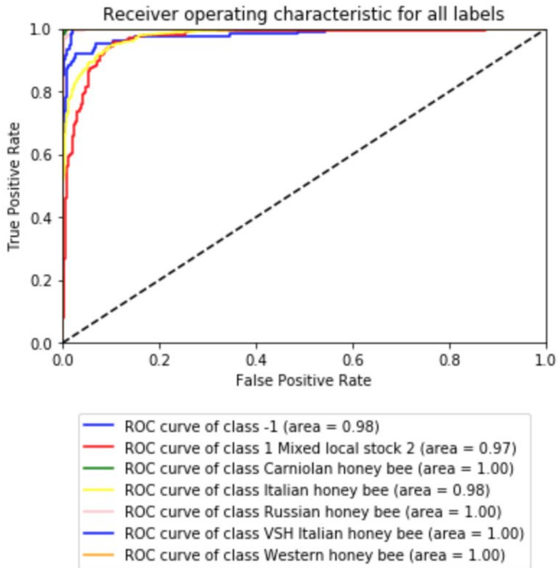
Since our data is imbalanced, also tried adding weights to sample, using custom weighted cross entropy loss and Focal loss function to handle it and compared each model. Meanwhile, due to the imbalance, the accuracy can not be counted as "accurate". F1 score is much more important. Adapting Cohen Kappa score to decide how much better the classifier is comparing with random guessing.

Conclusion:

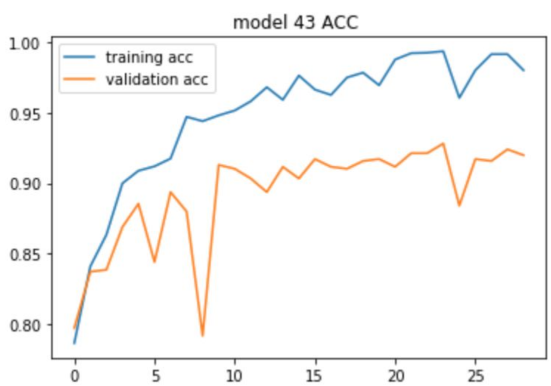
After trying, the best model are model43 and model43_f with focal loss according to f1 score, kappa score and time(model43 2s faster than model43_f). Then Compare them with ROC_AUC, model43_f did not outperform model43. Therefore focal loss seems not outperform CE in terms of traditional classification when dealing with this imbalanced data. Except for imbalance, another defect for our dataset is data size. We only have around 5000 images. The training size is only around 36000. In the future, I should try with keras image generator then use fit.generator to train the model. It can be used for image augmentation. Through keras image generator images can be flipped, shifted then act as new training samples. This way the training data could be increased.

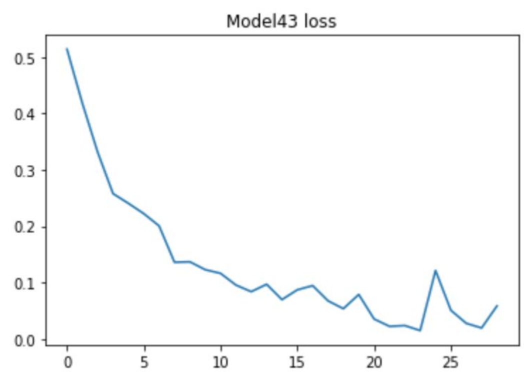
- ROC AUC Loss ACC

Model 43



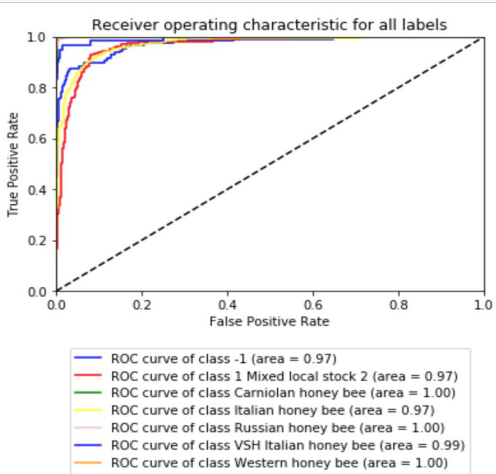

```
Text(0.5, 1.0, 'model 43 ACC ')
```



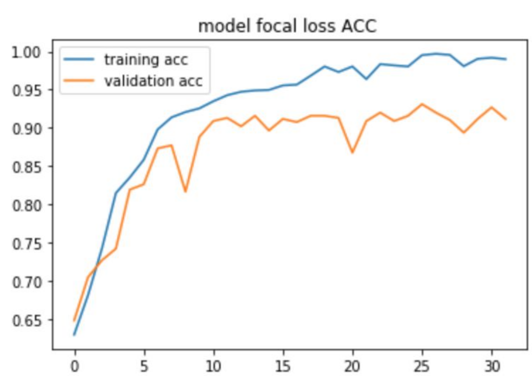


Model43_f

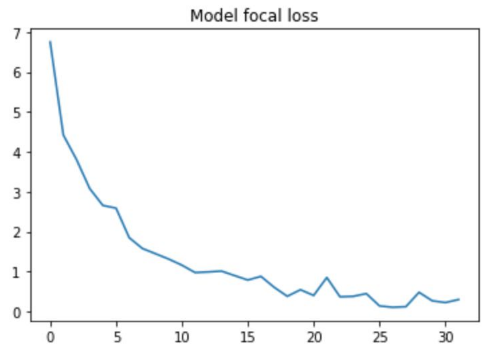
```
show_roc_auc(model43_f)
```



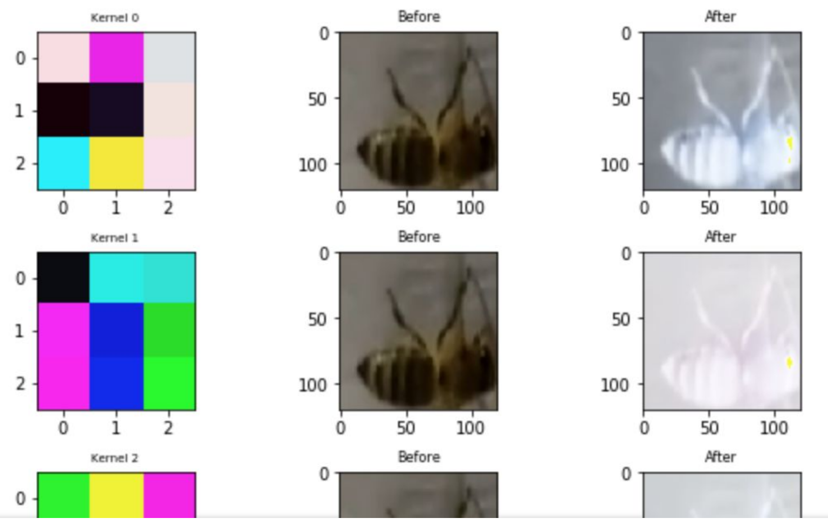

```
Text(0.5, 1.0, 'model focal loss ACC ')
```



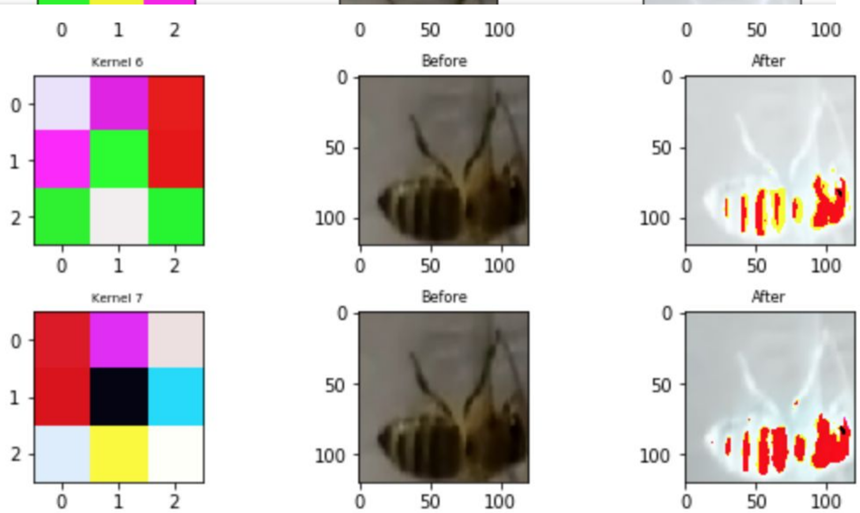
Text(0.5, 1.0, 'Model focal loss')

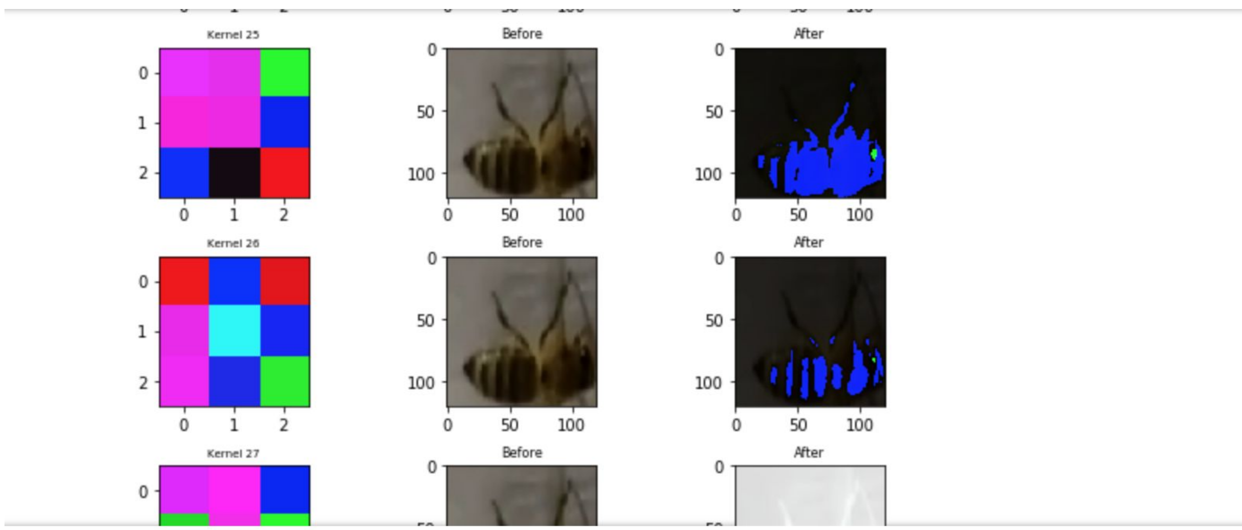


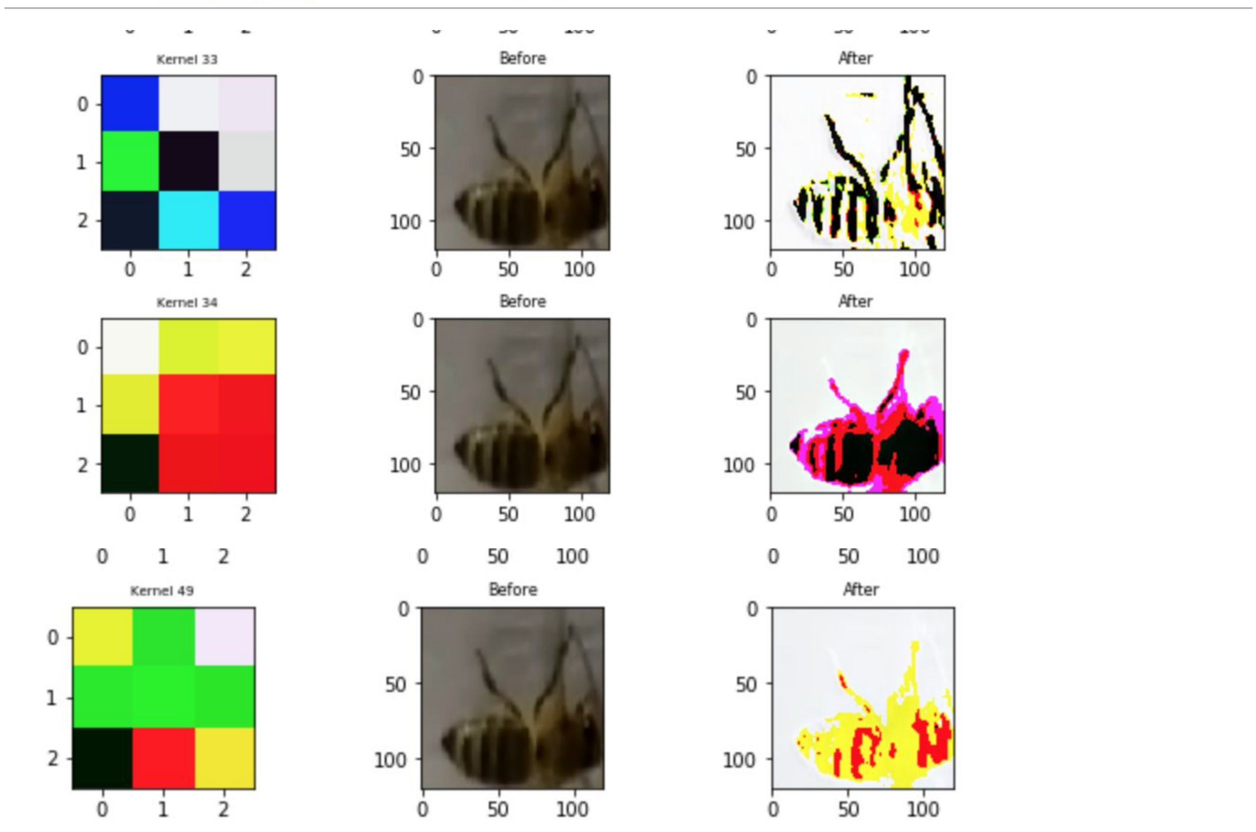
- **Kernel visualization for one image**



-



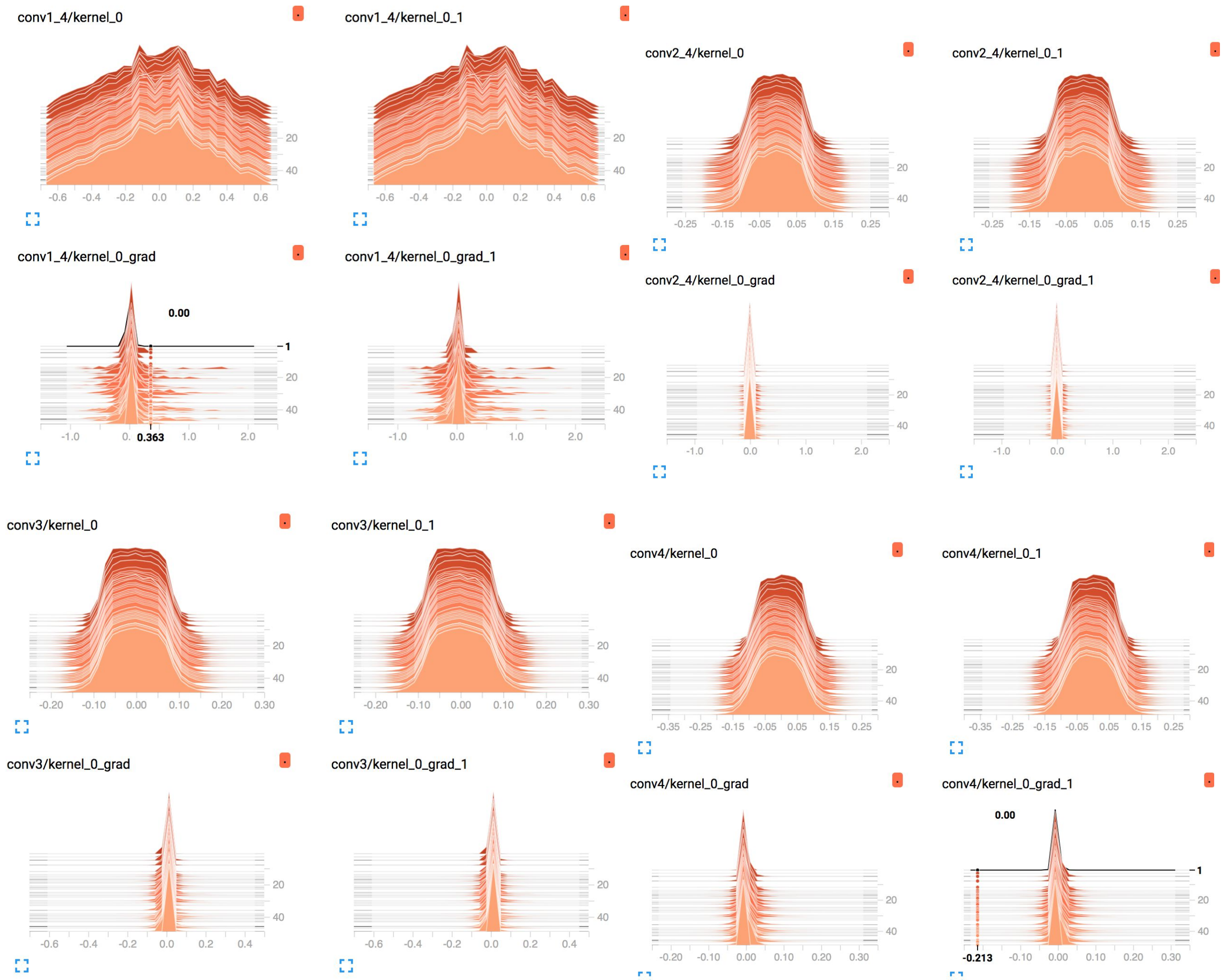




Most weights are applied on the back

- **Tensorboard for model 43**

  tensorboard --logdir ./Graph



Comparing the variance, std for kernels at each convolution layer, it did get trained.

Code copy:

```
[zhangruyuedeMacBook-Pro:Ruyue_Zhang_individual_project zhangruyue$ cd code
[zhangruyuedeMacBook-Pro:code zhangruyue$ tokei
-------------------------------------------------------------------------------
 Language            Files        Lines         Code     Comments       Blanks
-------------------------------------------------------------------------------
 Python                  1         1681          788          212          681
-------------------------------------------------------------------------------
 Total                   1         1681          788          212          681
-------------------------------------------------------------------------------
```

All code 788
img function: 9 all copy
Weighted loss function: 10 all copy
Focal loss: 8 all copy
Learningratescheduler: 7 modified 2
Show roc_auc: 18 modified 5
Visualization kernel: 19 modifiled 5

Own Code : 788-9-10-8-7-18-19=717
Copy rate:10%

Reference

Mkocabas, Muhammed. "Mkocabas/Focal-Loss-Keras." *GitHub*, 18 Mar. 2018, github.com/mkocabas/focal-loss-keras.

Shaoanlu. "Experiment: Applying Focal Loss on Cats-vs-Dogs Classification Task." *SALu*, 9 Nov. 2018, shaoanlu.wordpress.com/2017/08/16/applying-focal-loss-on-cats-vs-dogs-classification-task/.

Shaikh, Faizan, and Faizan. "Essentials of Deep Learning: Visualizing Convolutional Neural Networks in Python." *Analytics Vidhya*, 22 Mar. 2018, www.analyticsvidhya.com/blog/2018/03/essentials-of-deep-learning-visualizing-convolutional-neural-networks/.

Brownlee, Jason. "8 Tactics to Combat Imbalanced Classes in Your Machine Learning Dataset." *Machine Learning Mastery*, 7 June 2016, machinelearningmastery.com/tactics-to-combat-imbalanced-classes-in-your-machine-learning-dataset/.

Pukhov, Dmitry. "Honey Bee Health Detection with CNN." *RSNA Pneumonia Detection Challenge | Kaggle*, 2018, www.kaggle.com/dmitrypukhov/honey-bee-health-detection-with-cnn.

BigMoyan. "初始化方法." *初始化方法 - Keras中文文档*, keras-cn.readthedocs.io/en/latest/other/initializations/.

Godoy, Daniel. "Hyper-Parameters in Action! Part II - Weight Initializers." *Towards Data Science*, Towards Data Science, 18 June 2018, towardsdatascience.com/hyper-parameters-in-action-part-ii-weight-initializers-35aee1a28404.

Dahal, Paras. "Classification and Loss Evaluation - Softmax and Cross Entropy Loss." *DeepNotes*, 28 May 2017, deepnotes.io/softmax-crossentropy.

“Using sample_weight in Keras for Sequence Labelling.” *Stack Overflow*, 2018, stackoverflow.com/questions/48315094/using-sample-weight-in-keras-for-sequence-labelling.

Benshetler, Jeff . “Why does the accuracy decreases after I did data normalization in CNN model?” 26 Aug 2016, https://www.quora.com/Why-does-the-accuracy-decreases-after-I-did-data-normalization-in-CNN-model