[Team 12] Project C2: Deep Learning based Ensemble Model for Leaf Wilting Classification

Adithya Raghu Ganesh
Department of Computer Science
NC State University
araghug@ncsu.edu

Rakshita Ranganath
Department of Electrical Engineering
NC State University
rrangan2@ncsu.edu

Vivek Rathi
Department of Electrical Engineering
NC State University
vorathi@ncsu.edu

I. Introduction

Detection of water stress in plants is crucial in devising effective crop management strategies to improve the yield and quality of the crops cultivated. Leaf wilting and environmental data majorly contribute towards the estimation of water stress in plants. Manual inspection of the crops for detecting water stress is laborious and time-consuming. Hence, automating the entire process yields faster and accurate results. Considering only the image data for detecting the wilting in crops might not be efficient because of the variation due to manually annotated data, insufficient data resulting in class distribution imbalance, data overlap between neighbouring classes etc. Environmental factors such as humidity, temperature, wind speed, precipitation etc., can be considered in addition to the image data as they significantly influence plant wilting.

II. Related Work

Some of the earlier works were based on using the traditional image processing techniques by estimating leaf area to detect leaf wilting[1] and 2D Fourier transform applied to 3D image data of leaves[2]. Recent works based on machine learning and deep learning based techniques on RGB image data involves a sliding window based support vector regression[3], K-means clustering with ANN[4] and transfer learning models such as GoogleNet for wilt detection[5]. Various sensing and images techniques have been deployed in this regard which includes 3D laser scanning to estimate the leaf area index[6], thermal IR imaging[7], hyperspectral imaging[8] etc.

III. Data Augmentation and pre-processing

The data for plant wilting had 1275 images in total. This had a distribution of 488 images in Class 0, 329 images in class 1, 130 images in class 2, 131 images in Class 3 and 197 images in Class 4.

Since the distribution of the images provided were unequal, image augmentation was performed to obtain uniformly distributed data. Various data augmentation techniques were implemented such as horizontal flipping, lateral and vertical shifts, image rotation, incorporation of Gaussian noise, gamma correction with the parameter ranging from 0.25 to 2. In addition to these techniques, random cropping of the images was employed as better test accuracy was reported with this technique[9]. Data augmentation was performed only after splitting the data into training and validation to ensure that a completely unseen validation set is used.

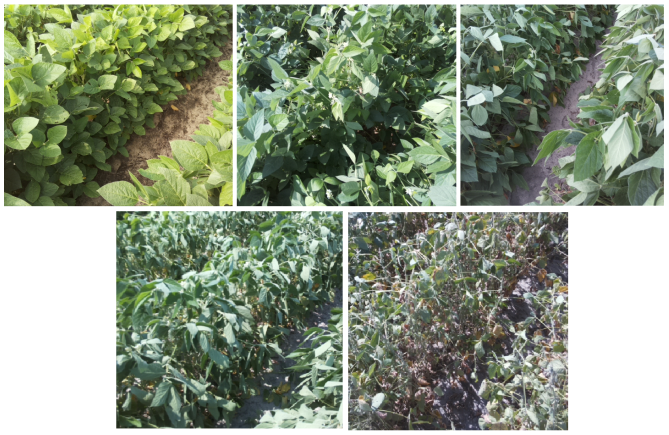


Fig. 1: Images for 5 classes

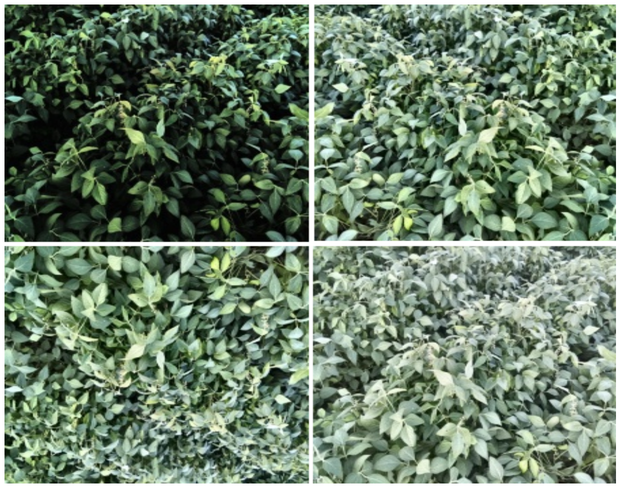


Fig. 2: Augmented Images

Incorporating data augmentation not only increased the number of images in each class but also added a bit of uncertainty on the data so that any model that trained on this data would not over-fit its parameters.
Due to the contrast and brightness variation amongst the images in the given data set, a variant of adaptive histogram equalization called Contrast Limited AHE (CLAHE) was used on the images to alleviate the difference in contrast levels.

IV. Methodology

Training a model from scratch is time consuming and a memory exhaustive process. In order to fasten the process of learning in cases where there is a scarcity of training images, transfer learning is incorporated. Transfer learning is a popular method used in computer vision to build both efficient and precise models for classification tasks. The main intuition behind transfer learning is to continue the training process from an architecture that has been pretrained for an alternate problem statement. Such a technique is proven to be accurate as the model picks up from the patterns learned for other datasets.

A pre-trained model is a model that has been previously trained on a large database of images such as imagenet. Such pretrained models help determine patterns more efficiently. We mimicked the ResNet and VGG16 network architectures for transfer learning.

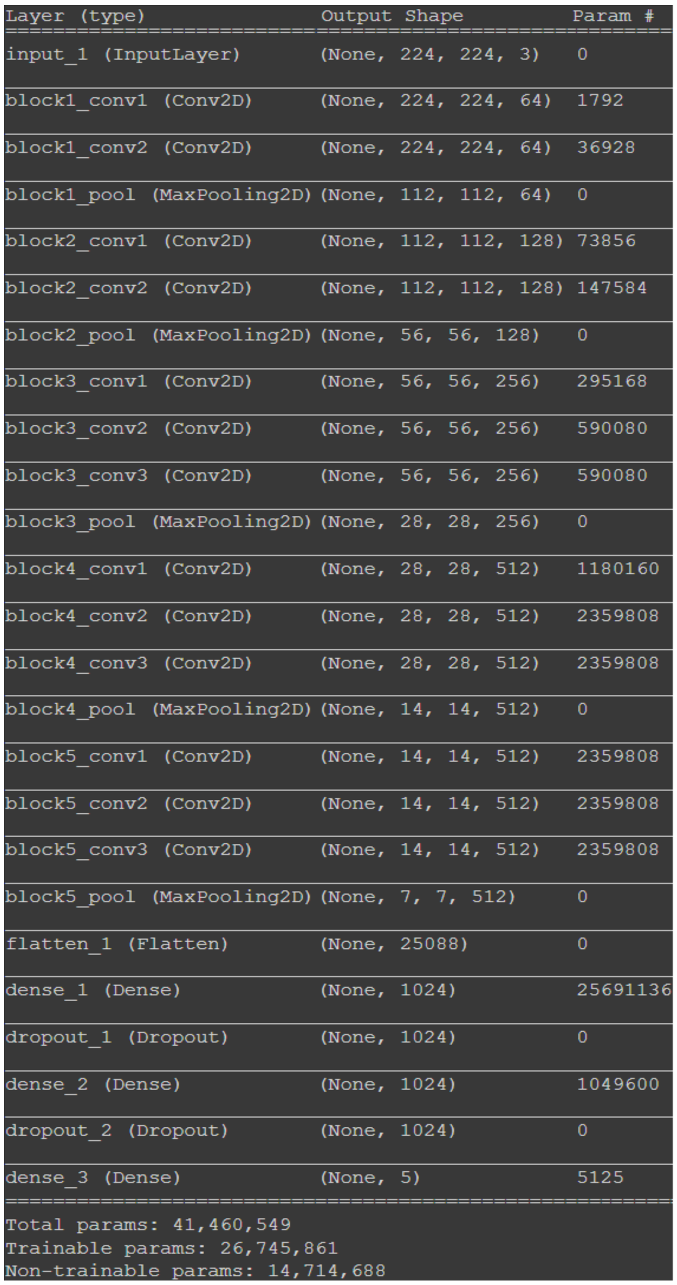
A. *Implemenation with ResNet architecture*

For image classification of wilted leaves where fine grained information is highly important, the loss of spatial information while downsampling may hinder the model from capturing essential information present in the image. It was seen that the ResNet50 Architecture exhibited mediocre performance due to its very slow convergence rate. Hence, this was modified to incorporate an additional loss, termed center loss. The main intuition behind using this novel loss was to increase inter-cluster distance and decrease the intra-cluster distance[10]. Upon implementation of the modified architecture in Keras[11] by making the last 30 layers of the ResNet50 trainable, a minor increase in the accuracy was observed. However, due to the spatial similarity present between the classes, it was concluded that other models architectures would perform better.

B. *Implementation with VGGNet*

From the last model, it can be seen that increasing the complexity of the model does not necessarily improve the models prediction. Thus, we decided to run our tests on another model that converges faster than ResNet. The VGG-16 architecture mainly consists of 13 Convolution layers, 5 max pooling layers and 3 fully connected layers. For the convolution layer, filter used has a small receptive field of 3x3 with a stride of 1 pixel. Max pooling is performed over a 2×2 pixel window with stride 2 and ReLU is used as the non-linearity for the activation layer.[12]

In the previous tests conducted, we found the model that imitated the VGG16 architecture[13] pre-trained on image-net along with two fully connected trainable layers to perform well on the validation data. With augmented data, it was ensured that each class contained around 500 images which was split into 450 images for training and 50 images for validation. Hyper-parameters for the network were tuned to achieve better performance and avoid over-fitting of the model. Types of optimizer, learning rate, number of fully connected layers and dropout were tweaked further by evaluating the training and validation metrics of model.



```
Layer (type)                 Output Shape              Param #
=================================================================
input_1 (InputLayer)         (None, 224, 224, 3)       0
_________________________________________________________________
block1_conv1 (Conv2D)        (None, 224, 224, 64)      1792
_________________________________________________________________
block1_conv2 (Conv2D)        (None, 224, 224, 64)      36928
_________________________________________________________________
block1_pool (MaxPooling2D)   (None, 112, 112, 64)      0
_________________________________________________________________
block2_conv1 (Conv2D)        (None, 112, 112, 128)     73856
_________________________________________________________________
block2_conv2 (Conv2D)        (None, 112, 112, 128)     147584
_________________________________________________________________
block2_pool (MaxPooling2D)   (None, 56, 56, 128)       0
_________________________________________________________________
block3_conv1 (Conv2D)        (None, 56, 56, 256)       295168
_________________________________________________________________
block3_conv2 (Conv2D)        (None, 56, 56, 256)       590080
_________________________________________________________________
block3_conv3 (Conv2D)        (None, 56, 56, 256)       590080
_________________________________________________________________
block3_pool (MaxPooling2D)   (None, 28, 28, 256)       0
_________________________________________________________________
block4_conv1 (Conv2D)        (None, 28, 28, 512)       1180160
_________________________________________________________________
block4_conv2 (Conv2D)        (None, 28, 28, 512)       2359808
_________________________________________________________________
block4_conv3 (Conv2D)        (None, 28, 28, 512)       2359808
_________________________________________________________________
block4_pool (MaxPooling2D)   (None, 14, 14, 512)       0
_________________________________________________________________
block5_conv1 (Conv2D)        (None, 14, 14, 512)       2359808
_________________________________________________________________
block5_conv2 (Conv2D)        (None, 14, 14, 512)       2359808
_________________________________________________________________
block5_conv3 (Conv2D)        (None, 14, 14, 512)       2359808
_________________________________________________________________
block5_pool (MaxPooling2D)   (None, 7, 7, 512)         0
_________________________________________________________________
flatten_1 (Flatten)          (None, 25088)             0
_________________________________________________________________
dense_1 (Dense)              (None, 1024)              25691136
_________________________________________________________________
dropout_1 (Dropout)          (None, 1024)              0
_________________________________________________________________
dense_2 (Dense)              (None, 1024)              1049600
_________________________________________________________________
dropout_2 (Dropout)          (None, 1024)              0
_________________________________________________________________
dense_3 (Dense)              (None, 5)                 5125
=================================================================
Total params: 41,460,549
Trainable params: 26,745,861
Non-trainable params: 14,714,688
```

Fig. 3: Architecture of VGG16

From the plots given in Fig 4, it can be seen that the network trains with a 30% increased accuracy when the learning rate of 0.0001 against 0.001 is used. From Fig 5, it is evident that the Adam optimiser outperforms others in giving better validation accuracy. Also, adding a dropout with 0.5 probability reduces the validation loss substantially giving it smoother transitions and is visualized in Fig 6.

With several tests involving hyperparameter tuning, a learning rate of 0.0001 coupled with the Adam optimizer function and dropout with 0.5 probability provided an accuracy of 60%.

However, considering distribution of the data for the obtained predictions, it was observed that more number of images were being classified as classes 0 and 4 in comparison to classes 1,2 and 3. Upon further examination, it was observed that slightly wilted images and unwilted images had high spatial similarity just like completely wilted and almost fully wilted images.

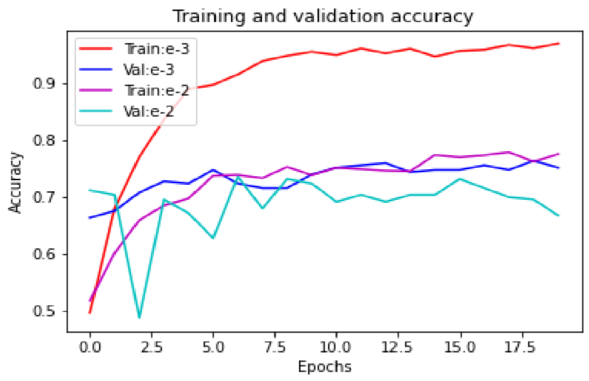


Fig. 4: Learning Rate Selection

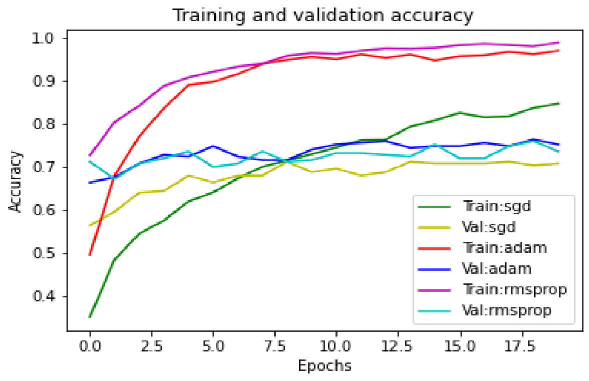


Fig. 5: Optimiser Selection

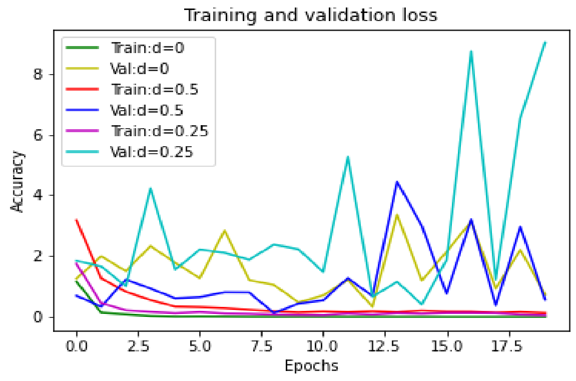
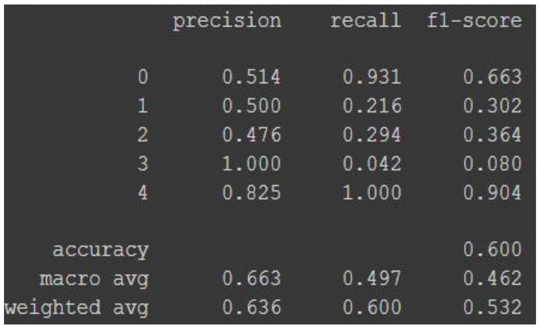


Fig. 6: Dropout Selection

Based on the above data, we developed a novel model that was initially trained on images belonging to classes 1,2 and 3. We then retrained the same model on all 5 classes to create a model that gave a higher priority to classify spatially similar classes. It was noticed that such a model did indeed decrease the mean squared error (MSE) of the validation data but it also ended up classifying images in Classes 1-3 more frequently. Upon further examination of the results obtained from the second model, it was inferred that there has to be a balance established between the two models.

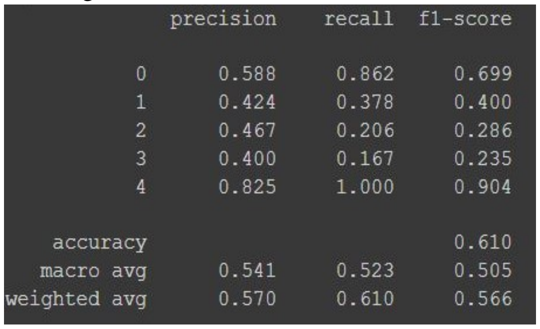


```
              precision    recall  f1-score

           0      0.514     0.931     0.663
           1      0.500     0.216     0.302
           2      0.476     0.294     0.364
           3      1.000     0.042     0.080
           4      0.825     1.000     0.904

    accuracy                          0.600
   macro avg      0.663     0.497     0.462
weighted avg      0.636     0.600     0.532
```

Fig. 7: Evaluation Metrics for VGG16



```
              precision    recall  f1-score

           0      0.588     0.862     0.699
           1      0.424     0.378     0.400
           2      0.467     0.206     0.286
           3      0.400     0.167     0.235
           4      0.825     1.000     0.904

    accuracy                          0.610
   macro avg      0.541     0.523     0.505
weighted avg      0.570     0.610     0.566
```

Fig. 8: Evaluation Metrics for Ensemble Model

This gave rise to the idea of creating an ensemble model for the classification problem. Here, the mean of the classification results was taken from both the models whenever model 1 gave a prediction as class 0 or 4 and the output was taken from model 2 alone when the prediction from model 1 belonged to classes 1,2 and 3.

We felt that since one model prioritized classes 1,2 and 3 and the other struck a balance between all 5 classes, such an ensemble model for this problem statement was ideal. The corresponding metrics have been presented in Fig. 7 and 8.

V. Evaluation

After several iterations of hyperparameter tuning and architecture modifications, the different performance metrics have been compared. Apart from the accuracy of the model, the top-2 accuracy is also important. Due to the high spatial similarity between the classes and the availability of small training data, we need to ensure a very low MSE. The top 2 accuracy takes this into consideration and gives us perspective on how close the model was in predicting the true class. Thus, from the top-2 accuracy, we get a better intuition about a models performance.

Model	Top 1 Acc.	Top 2 Acc.
VGG-16	60%	74%
VGG-16 *	60%	71.5%
Ensemble Model	**61%**	**80%**
ResNet50	42%	51%

Table 1. Top 1 and Top 2 accuracy for the models

It is evident from the results that an ensemble architecture with a uniquely weighted classification is ideal for such a use case. This architecture was able to classify the images despite the high ambiguity present in the data and the robustness of this model could be attributed to the way the weights were trained for each class.

VI. Extra credit using weather data

Due to the shortcomings while dealing with just the image data as mentioned above, environmental factors such as temperature, humidity, shortwave radiation etc. - 12 factors in total were taken into account. Out of the 1275 available images, 1175 images were considered for training with a 0.8 split and 100 images for testing respectively.

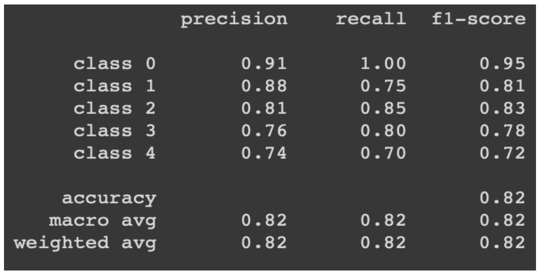
A. Model specification

Various techniques such as Random forests classifier, Gaussian Naive Bayes Classifier and Support Vector Classifier with RBF kernel were used for training and they gave accuracies of 60%, 44% and 63% respectively.The main challenge in obtaining higher accuracy with SVM involved kernel selection in higher dimensional space that provided the best separation when the classes are linearly inseparable. Hence, an ensemble approach was considered as it could learn the non-linear and implicit relationships. With the inherent advantages of boosting techniques, Extreme Gradient Boosting (XGBoost) was used for its high execution speed and better model performance.

XGBoost model[14] was implemented with a maximum depth of 4 for each tree, a training step of 0.05 for each iteration and a total of 20 iterations with softprob as the objective function to obtain prediction probabilities.

B. Evaluation

An accuracy of 72% was obtained on the numerical test data with XGBoost model. The CNN architecture using VGG16 network on image data provided an accuracy of 79%. However, in order to improve the previous results, a final ensemble of the XGBoost and CNN models was created. The test data distributions for each of the two models were compared and the best classifier for each class was determined. From the insights generated, it can be seen that VGG16 model classifies classes 0, 1 & 3 effectively and the XGBoost performs better with classes 2 & 4. Considering these statistics, the ensemble model chooses optimum classifier amongst the two giving an improved accuracy of 82%. Precision, recall and F1 score for each class is shown in Fig 9.


```
              precision    recall  f1-score

     class 0       0.91      1.00      0.95
     class 1       0.88      0.75      0.81
     class 2       0.81      0.85      0.83
     class 3       0.76      0.80      0.78
     class 4       0.74      0.70      0.72

    accuracy                           0.82
   macro avg       0.82      0.82      0.82
weighted avg       0.82      0.82      0.82
```

Fig. 9: Evaluation metrics for Ensemble classifier

VII. Results

The VGG16 network that trained on both spatial and textual data gave an accuracy of 79% and the XGBoost gave an accuracy of 72% for the same data. Our novel ensemble model gave an accuracy of 82%. The predictions obtained for each class from the above mentioned models are tabulated in Table 2.

Class	VGG16	XGBoost	Combined model
0	26	40	20
1	24	15	15
2	12	19	17
3	24	7	16
4	14	19	14
Accuracy	79%	72%	**82%**

Table 2. Distribution of correct predictions for each class

VIII. Conclusion

This report proposes an ensemble model for leaf wilting classification and determines a way to intuitively incorporate the VGG-16 architecture to improve the base line predictions. We have also implemented a combined model that takes into account the weather data for extra credit. Both the proposed architectures where implemented in Python using the GPU provided in Google Colab notebooks.

References

[1] C. J. Paull E. Nyakwende and J. G. Atherton. Non-destructive determination of leaf area in tomato plants using image processing. *Journal of Horticultural Science*, 72(2):255–262, 1997.

[2] Y. Sun X. Cai, Y. Zhao, et al. Smart detection of leaf wilting by 3d image processing and 2d fourier transform. *Computers and Electronics in Agriculture*, 90:68 – 75, 2013.

[3] Yukimasa Kaneda, Shun Shibata, and Hiroshi Mineno. Multi-modal sliding window-based support vector regression for predicting plant water stress. *Knowledge-Based Systems*, 134:135 – 148, 2017.

[4] S. Veni R. Anand and J. Aravinth. An application of image processing techniques for detection of diseases on brinjal leaves using k-means clustering method. In *2016 International Conference on Recent Trends in Information Technology (ICRTIT)*, pages 1–6, 2016.

[5] S. I. Hassan L. M. Dang et al. Uav based wilt detection system via convolutional neural network. *Sustainable Computing: Informatics and Systems*, 90:68 – 75, 2018.

[6] Katrine Heinsvig Kjær and Carl-Otto Ottosen. 3d laser triangulation, a simple and robust method for automated growth determination of crop plants in challenging environments. *Sensors*, 15:2, 2015.

[7] Rocío Calderón, Juan A Navas-Cortés, and Pablo J Zarco-Tejada. Early detection and quantification of verticillium wilt in olive using hyperspectral and thermal imagery over large areas. *Remote Sensing*, 7(5):5584–5610, 2015.

[8] Craig R Yendrek, Tiago Tomaz, Christopher M Montes, Youyuan Cao, Alison M Morse, Patrick J Brown, Lauren M McIntyre, Andrew DB Leakey, and Elizabeth A Ainsworth. High-throughput phenotyping of maize leaf physiological and biochemical traits using hyperspectral reflectance. *Plant physiology*, 173(1):614–626, 2017.

[9] R. Takahashi, T. Matsubara, and K. Uehara. Data augmentation using random image cropping and patching for deep cnns. *CoRR*, abs/1811.09030, 2018.

[10] Yandong Wen, Kaipeng Zhang, Zhifeng Li, and Yu Qiao. A discriminative feature learning approach for deep face recognition. In *European conference on computer vision*, pages 499–515. Springer, 2016.

[11] *https://keras.io/applications/resnet*.

[12] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *CoRR*, abs/1409.1556, 2015.

[13] *https://keras.io/applications/vgg16*.

[14] *https://xgboost.readthedocs.io/en/latest/python/python*$_intro.html$.