Animal classification from greyscale images Kaggle competition

1 Information about the Competition

1.1 Team members

The team name in Kagggle is **Outliers**. Our team consists of the following three members:

- Ali akbar Sabzi : ali-akbar.sabzi@polymtl.ca, 2078921.
  Student at Polytechnique Montreal
- Hazem Barka : hazem.barka.1@ens.etsmtl.ca, 2174747.Student at ETS
- Mohamed Chiheb Ben Nasr : mohamed-chiheb.ben-nasr.1@ens.etsmtl.ca, 2174815.
  Student at ETS

1.2 Framework of the competition

This kaggle competition is within the framework of the course INF8245E - Machine Learning. The goal of the competition is to design a machine learning algorithm to predict the animal class from 11 different classes. The length of the provided training set is 11, 887 uni-channel images with size (96,99). The cardinality of the test set is 17, 831. The evaluation metric is the F1-score although other metrics such as the accuracy may be employed in this report.

2 Feature design

In this project, two main approaches were adopted for feature extraction. In each of these approaches, we will specify the preprocessing, the intuition behind the algorithms and the selected features. It is worth noting that only a brief description will be provided for the techniques due to the space limitations.

2.1 Computer vision based feature selection

In this subsection, we will provide the first employed method. In fact, it has been shown that computer vision based feature extraction is very efficient in image classification and still offers competitive results compared to deep learning applications Wang et al. (2017), Rançon et al. (2019a), Rançon et al. (2019b). Such works proves that non deep learning based feature extraction methods such as SIFT (Scale Invariant Feature Transformation) still have a place in the realm of image classification even in the era of Deep Learning. The particular attractiveness of such approaches is that they do not present a black box as to their functioning. It is worth mentioning that SIFT was patented until late march 2020 and is currently available for use (https://patents.google.com/patent/US6711293).

2.1.1 SIFT

The main idea behind SIFT algorithm is to detect interesting key-points (pixels), while making sure that the selected key-points are invariable to most of the known computer vision challenges. The list below provides such challenges and works that proves that SIFT algorithms can easily tackle such challenges:

- **Viewpoint:** some object can be detected wrongfully from different viewpoints Chen & Shang (2016).
- **Deformation:** some objects may present deformations that makes it harder for systems to detect them Zheng & Qian (2012).
- **Occlusion:** represents the case where an object masks another one Zhou et al. (2009).

- **Illumination:** represents the challenge induced by the change of illumination in the image Wang et al. (2015).
- **Texture:** the textures are not easily detectable using SIFT and presents one of the challenges of such algorithm.
- **Intraclass variation:** also SIFT struggles in this particular challenge.

The majority of the aforementioned challenges are present in the provided dataset of this competition, for example, the presence of ’Cats’ and ’Big cats’ classes is a typical case of the *Intraclass variation* challenge. Other challenges such Texture are also present in the provided dataset. As illustrated in figure 1, all the computer vision challenges are present in the provided dataset. As such, this motivates the use of SIFT features. Another motivation of the use of the SIFT algorithm is its scale invariance. SIFT is conceptualized as to be invariant to the difference in the scale. This is achieved by the use of four different steps. A brief explanation of SIFT algorithm is provided in the appendix.

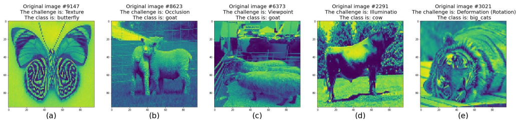

Figure 1: Examples of challenges present in the dataset. (a) :texture, (b): occlusion, (c): viewpoint, (d): illumination, (e): deformation (rotation)

2.1.2 Mathematical morphology

Mathematical morphology are a common approach used in computer vision in order to analyse the structures of geometric objects. In particular, certain combinations of operands (in particular erosion followed by dilation) helps achieve a closing Tang et al. (2012). Closing, as its name implies helps close certain geometrical shapes. This was used in our work in order to create a mask that represents the shapes of the objects. This is similar to the cropping but with more refined edges and no resizing option.

2.1.3 Resulting detection

After applying SIFT and several iteration (20) of closing in the binary image created by the localized descriptors of SIFT, we obtained a new feature of the images that we called the masked images. This feature provides a localized and specific description of the geometry of the animal and can be used as a mask to provide more attention to certain areas on the image also called Region of Interests (ROI). Furthermore, a cropped images was created from this particular feature. This cropped image provides a zoomed image of the detected ROI is the previous step and thus provides our third extracted feature. Figure 2 illustrates the aforementioned explanation. The second row of images provides the created SIFT keypoints described previously. The third row presents the masked images (closing) and the last row provides the cropped and zoomed images.

To conclude, a combined approach of SIFT and mathematical morphology helped us create two additional channels to use for the classification of the animals.

2.2 Deep learning based feature extraction

For the deep learning part, several architectures were adopted to achieve the optimal feature selection. In particular, both Resnet18 and densenet121 CNN architectures were employed in this work. First, we will tackle the different data augmentation techniques introduced in this section.

2.2.1 Data augmentation

Three different techniques were introduced in our work to augment the data. First, we introduced **resizing** where we randomly changed the size of the images from (96,96) to (224,224). This helps

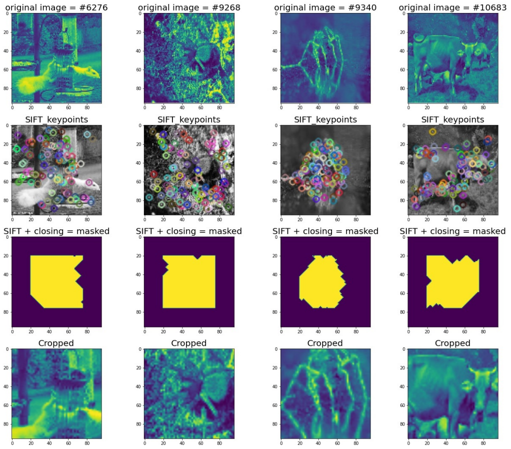

Figure 2: The created channels

augment the data by introducing more samples and creates a certain invariance by scale (same to the one introduced in SIFT). The second utilized technique is **flipping**. During this phase, the train images were randomly flipped (rotation of 180 degrees) in order to achieve a certain invariance by flips as well as augment the data size. Finally to push this even further, **random rotations** with fixed degrees were introduced to achieve invariance by rotation.

2.2.2 Data preprocessing

The normalization of the data was introduced by dividing the pixels of the images by a factor of 255.

2.2.3 Resnet

First introduced in He et al. (2015), the main idea behind the Resnet18 architecture is to allow the network to model a residual function instead of the actual label of the data. More details about this architecture is provided in Appendix 2.

2.2.4 Densenet

As introduced in Huang et al. (2016), the major difference between Densenet and Resent architecture resides in the connections between the convolutional blocks. More details about this model are provided in Appendix 2.

2.2.5 Xception

First introduced in Chollet (2016), the main idea behind the Xception algorithm is to employ depthwise separable convolutional layers. The hypothesis is that the convolutional layers cross-channel wise (illustrated in figure 2) and spatial wise can be decoupled.

3 Algorithms

In this section, we will discuss the utilized algorithms for the prediction.

3.1 Using Resnet18 features

We used the features extracted from Resnet18 detailed in the previous sections with different classification models in order to obtain optimal results.

3.1.1 Artificial Neural Network

Two different models were created using ANN architectures. In particular, the use of softmax and log softmax was employed. It is worth mentioning that the use of log softmax provides better numerical performance and gradient optimization. It is one of the **optimization techniques** utilized in this project. The employed loss function is sparse categorical cross-entropy and the used optimized is 'adam' since it provided the best learning for the model (compared to SGD). The activation function is 'Relu' and the learning rate is 0.001.

3.1.2 Random forest classifier

Apart from the ANN classifiers used, we employed a random forest classifier with the features extracted from Resnet18. The hyperparameters picked for this model are discussed in the methodology section.

3.2 Using Densenet121 features

Only ANN architecture was used as a classifier for the generated Densenet121 features. We employed similar loss, optimizer, activation function, learning rate and architecture as the previous case.

3.3 Using the computer vision channels

After the creation of 3 channels in the computer vision feature extraction, we chose to use the three channels on a predefined architecture, in particular the chosen architecture is Xception Chollet (2016). This architecture is the developed version of the Inception architecture and is based on depthwise separable convolution layers. This architecture, as opposed to Resnet18 and Densenet121 was fully used for classification and not only feature extraction.

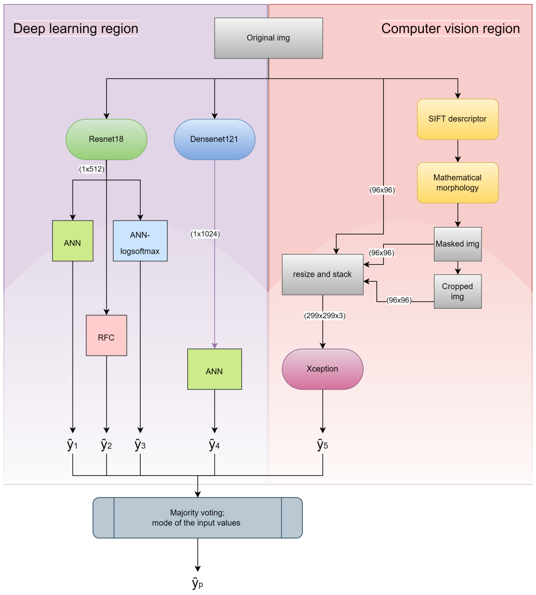

Figure 3: Final solution overview

Figure 3 illustrates the final prediction methodology employed in this project.

4 Methodology

In this section, we will provide the methodology of our work. In particular, three different subsections will be discussed iterating over the different steps our work adopted. Mainly feature extraction using SIFT, using elaborated architectures and the final classification algorithm.

4.1 Baseline model

For the baseline model we have performed nonlinear SVM and Gradient boosting. Since the results of SVM were better than gradient boosting we report only SVM results in this report. The baseline model was performed without any feature selection; the only preprocessing performed was flattening the dataset and division by 255. Below table shows its performance with and without hyperparameter tuning on the training, validation and test dataset splitted from the original training dataset.

	Train	Validation	Test
Default parameters	0.571	0.253	0.256
Hyperparameter tuning	0.999	1	0.37

Table 1: F1-scores for the baseline SVM classifier

Please note that the hyperparameter used for tuning was ‘C’ (Regularization parameter) which was investigated in the range of 1 to 100 with steps equal to 20. F1-score metric for test dataset on kaggle submission gave the result equal to 0.35.

4.2 Computer vision methodology

Concerning the computer vision based algorithms, no splitting was done. This is because of the nature of the algorithms that does not require any prior training. The hyperparameters of the SIFT algorithm were fine tuned as to provide the best mask and cropped images. We mention particularly, **nfeatures** which is the number of features extracted from the image, **nOctaveLayers** which represents the number of octave layers (how many times we are scaling the images to obtain scale invariance), **sigma** which is the standard deviation of the used Gaussian which rules over the blurring factor of the algorithm and finally **contrastThreshold** which is the selectivity of the SIFT model. Concerning the mathematical morphology operator, the only hyperparameter is the number of times the closing is going to be applied. This was also manually selected as to provide closed enough geometries the selected value is 20.

4.3 Deep learning methodology

A classical train-validation and test split was employed in this section. The ratios of **0.9** for train, **0.05** for validation and **0.05** for test was employed. The training data was employed to train the Resnet and Densenet architectures and thus extract the needed features. The validation set was utilized in order to fine-tune the hyperparameters of the network (mainly the epochs and the batch size). **Early stopping** was utilized in order to obtain the best balance between goodness of fit and generalization. The Deep learning optimize utilized is **’Adam’** optimizer and the activation functions used are **’relu’**. For the case of the training of Resnet18 both **softmax** and the **log of softmax** were utilized to extract the features. Concerning the regularisation methodology, the use of **dropout layers** and **weight decay** is incorporated in the Densenet, Resnet and Xception architectures which are the most common regularisation methods in deep learning.

4.4 Classification methodology

The decision based classifier utilized in this work is random forest classifier. The same train-validation-test split was employed in this section and it was done from the feature extraction step. The trained model was than evaluated on the validation set in order to obtain the best hyperparameter. The choice of hyperparameters was done manually. The following hyperparameters have proven to show the best results: **number of estimators:** 100, **min samples split:** 2, **min samples leaf:** 1. The fine tuning of the hyperparameters helped us obtain a decent outcome of the model while tweaking the regularization parameters (hyperparameters of the model).

5 Results

In this section, we will provide an overview of the utilized methods and their respective results.

5.1 Resnet18, Densenet121 and Xception classifiers

The learning curve of the Resnet18, Densenet121 and Xception classifiers is provided in the appendix 2 figures 7, 8 and 9 respectively, Note that : Xception classifier is used on the channels created from the computer vision methodology.

5.2 Random forest classifier

The random forest classifier using the features extracted from the Resnet18 architecture showed promising results. This model was thus included in our overall submission. Table 2 shows the results of the random fores classifier.

metric	Train	Validation	Test
accuracy	1	0.81	0.79
f1-score	1	0.79	0.8

Table 2: Accuracy and F1-scores for the random forest classifier

5.3 Majority voting

After gathering five different classifiers with close f1-scores, the final model is a majority voting of these models. The class which has the most votes is selected. The results are shown in table 3

Model	f1-Train	f1-validation	f1-competition
Resnet18-ANN-softmax	1	0.80	0.81
Resnet18-ANN-logsoftmax	1	0.805	0.81
Resnet18-RFC	1	0.81	0.79
Densenet121-ANN-softmax	1	0.80	0.78
Computervision-Xception-ANN-softmax	0.99	0.78	0.79
final model submission	0.99	–	**0.836**

Table 3: Accuracy and F1-scores for the random forest classifier, final submitted result in bold

6 Discussion

In this work, we provided two main orientations of our work. For the computer vision based results, we already mentioned that the SIFT model lacks when it comes to texture detection and given that such samples exist in the provided data one of the fields of work can be to improve on this fact. Some features such as Local Binary Pattern can be utilized for such results. Concerning the Deep learning part, more data augmentation based on GAN architectures for example can be utilized. Also, attention mechanisms in CNN architectures can help further improve the obtained results.

7 Statement of Contributions

This work was the result of team working and collaboration of the members of the team. In particular, Hazem provided the results of the deep learning based models, mainly Densenet121, Resnet18 and RFC classifiers, Chiheb provided the computer vision based methodology as well as the Xception model and Ali provided the vgg19 model and useful insights overall the projects as well as helped in defining the problematic and baseline models.

A Appendix 1: More on SIFT algorithm

As explained in the second section, the SIFT algorithm is conceptualized in order to be scale invariant and immune to most of the computer vision challenges. This is achieved by four different steps.

The first step is to provide a scale space (generate different octaves) combined with a Difference of Gaussian (DoG) and the calculation of the gradient (between the neighbors in the same octave and cross octaves) provides an eliminatory key-point detection mechanism. This achieves a feature extraction step with interest points that are scale invariant (different octaves) and resolution invariant (Gaussian blurring). This process is illustrated in figure 5.

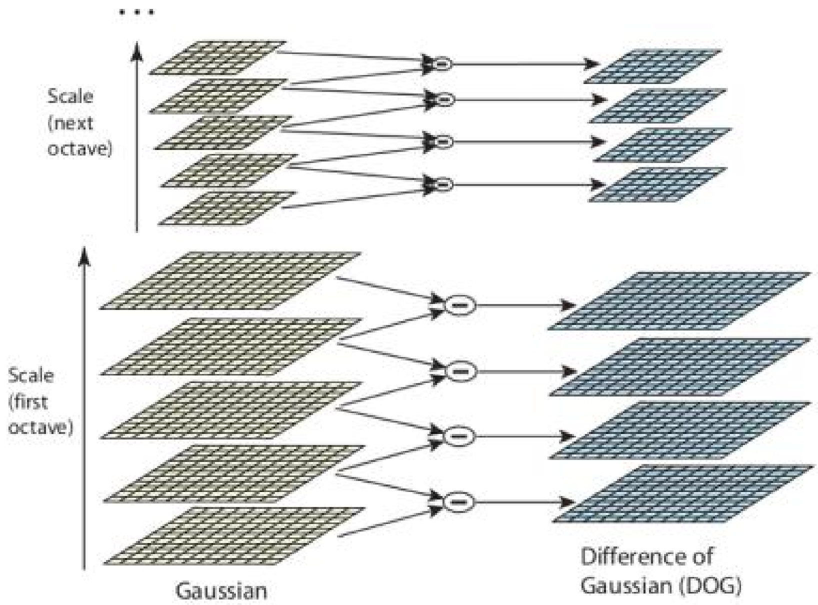

Figure 4: Difference of Gaussians illustrated in LoweDavid (2004)

Second, refining the detected keypoints. At this step, three different types of keypoints are detected: edges, flat and accurate keypoints. The flat areas can be filtered by their intensity, for the edges, the combined use of Taylor expansion and the eigen values of the Hessian matrix with a threshold selection provides an accurate filter for the edges.

Third, the orientation and the scale of the gradient of detected keypoints is computed. The values as well as the bins of a created histogram (each bin 10 degrees) are taken if they surpass certain threshold which thus creates a certain rotation invarince of the keypoint.

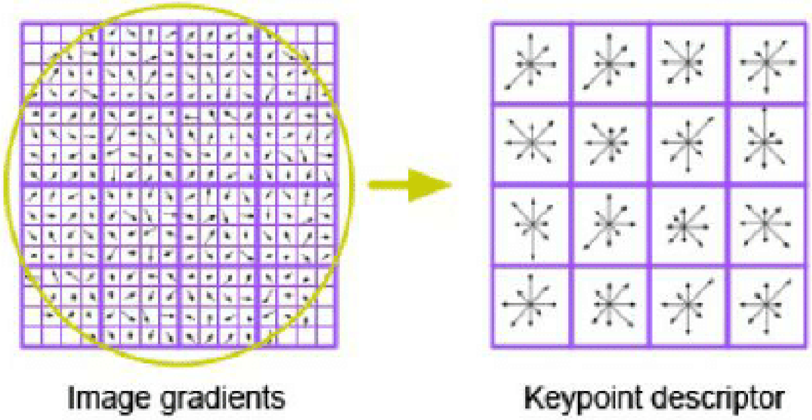

Figure 5: Keypoints extraction illustrated in Adel et al. (2014)

The last step is to calculate the keypoint descriptor. A neighborhood of (4x4) pixels is chosen and the keypoint descriptors represents the histogram (scale + orientation) of the keypoint with respect to the selected neighbor. This is typically calculated for an 8 bin histogram, so the resulting descriptor is of dimension 4x4x8 = 128. This is illustrated in figure **??**

In our work, we only used the first three steps. In fact, we found more interest in the calculation of the SIFT to localize the keypoints. However, the values of the descriptors were discarded for more advanced features.

A Appendix 2 : Deep learning models

A.1 Resnet18

As mentioned previously, the Resnet18 architecture was first introduced in He et al. (2015), the main idea behind the Resnet18 architecture is to allow the network to model a residual function instead of the actual label of the data. This helps achieve what the author describes as ”identity mapping” which refers to achieving a network where the depth doesn’t ruin the accuracy of the model (in the classic case, if a shallow network achieves a certain accuracy a deeper one, counter intuitively, doesn’t achieve similar results), thus the idea of introducing shortcuts in the network. In this work, we used to architecture of Resnet18 which refers to the number of convolutional blocks employed in this architecture, we than trained it on the introduced data and thus extracted 512 feature per image.

A.2 Densenet121

As previously mentioned, Densenet architecture was first described in Huang et al. (2016), the major difference between Densenet and Resent architecture resides in the connections between the convolutional blocks. More details about this model are provided in Appendix 2. . Referred to by the author by dense blocks, these blocks introduce interconnection between not only consecutive convolutional nodes, but also between the following layers. Densenet121 refers to the architecture that provides four dense blocks with intermediate transition layers. The major difference between the other Densenet architectures is in the choice of the last two dense blocs as they present respectively 24 and 16 convolutional blocks. 1024 features were extracted. Figure 6 illustrates the different architectures.

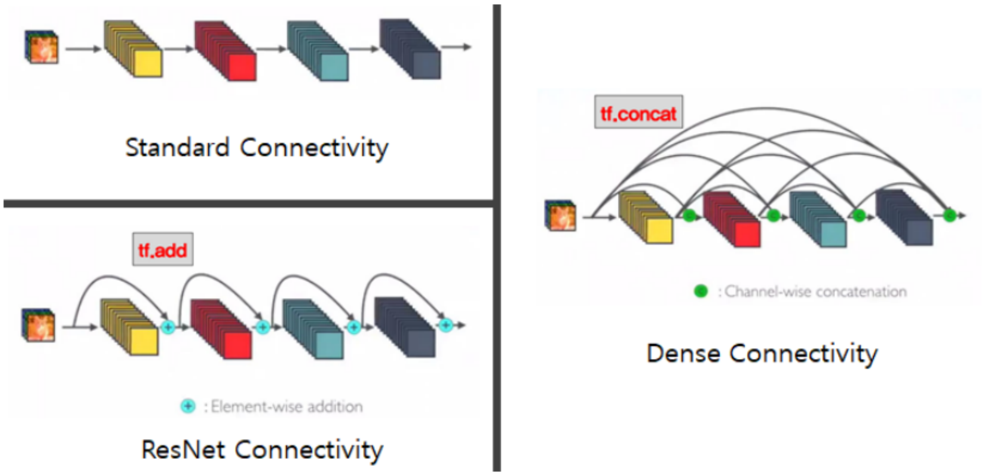

Figure 6: Comparison between the classical and Resnet and Densenet architectures Abai & Rajmalwar (2019)

A Appendix 3 : Learning curves

The learning curves are included in this appendix. In particular, the three architectures used in this work, mainly the Xception, Densenet and Resnet. The loss implemented is the sparse categorical cross entropy. The learning curves are evaluated by accuracy and f1-score for both Densenet and Resnet architecture. These models were later utilized in majority voting in order to boost the performance of the last model.

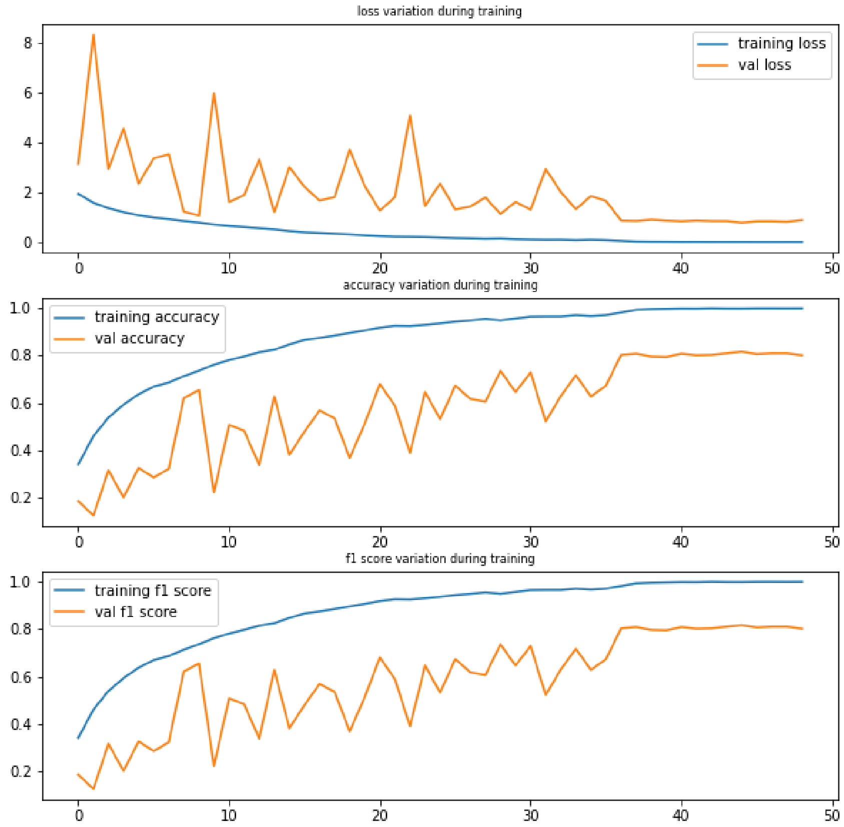

Figure 7: Learning curves of Resnet18 model

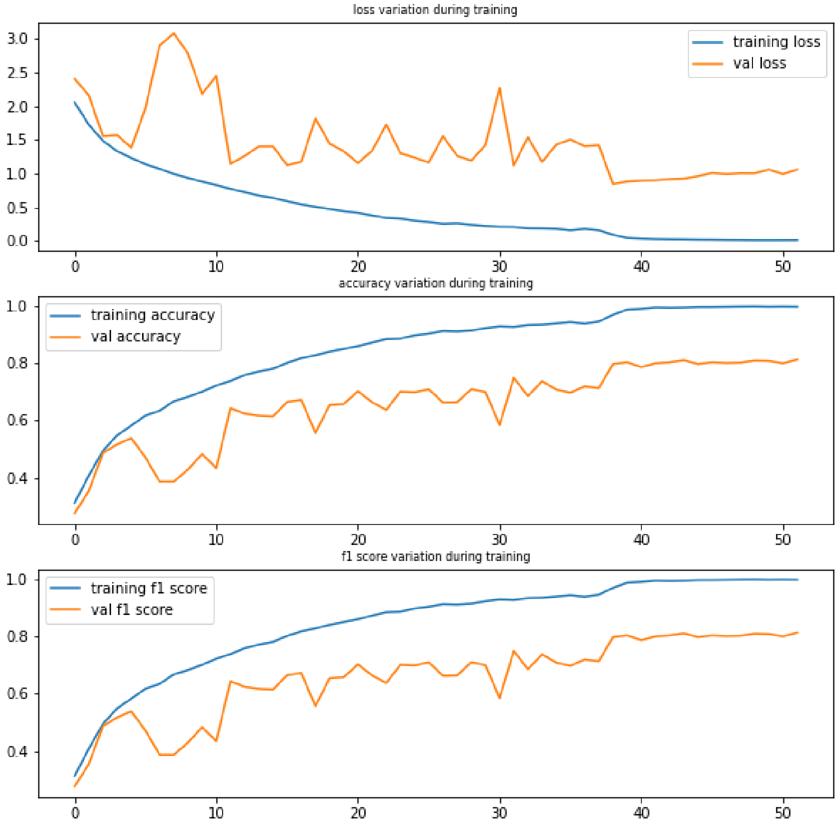

Figure 8: Learning curves of Densenet121 model

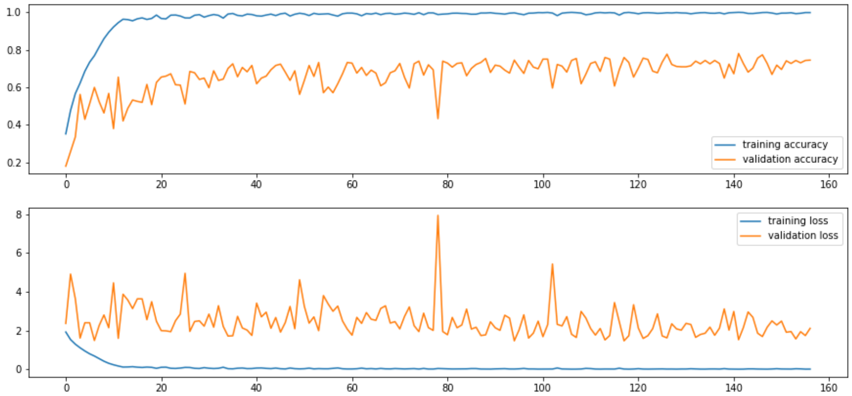

Figure 9: Learning curves of Xception model

REFERENCES

Zoheb Abai and Nishad Rajmalwar. Densenet models for tiny imagenet classification. *CoRR*, abs/1904.10429, 2019. URL `http://arxiv.org/abs/1904.10429`.

Ebtsam Adel, Mohammed Elmogy, and Hazem El-Bakry. Image stitching based on feature extraction techniques: A survey. *International Journal of Computer Applications*, 99:1–8, 08 2014. doi: 10.5120/17374-7818.

Yong Chen and Lei Shang. Improved sift image registration algorithm on characteristic statistical distributions and consistency constraint. *Optik*, 127(2):900–911, 2016.

François Chollet. Xception: Deep learning with depthwise separable convolutions. *CoRR*, abs/1610.02357, 2016. URL `http://arxiv.org/abs/1610.02357`.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. *CoRR*, abs/1512.03385, 2015. URL `http://arxiv.org/abs/1512.03385`.

Gao Huang, Zhuang Liu, and Kilian Q. Weinberger. Densely connected convolutional networks. *CoRR*, abs/1608.06993, 2016. URL `http://arxiv.org/abs/1608.06993`.

G LoweDavid. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 2004.

Florian Rançon, Lionel Bombrun, Barna Keresztes, and Christian Germain. Comparison of sift encoded and deep learning features for the classification and detection of esca disease in bordeaux vineyards. *Remote Sensing*, 11(1):1, 2019a.

Florian Rançon, Lionel Bombrun, Barna Keresztes, and Christian Germain. Comparison of sift encoded and deep learning features for the classification and detection of esca disease in bordeaux vineyards. *Remote Sensing*, 11(1):1, 2019b.

Jing-Tian Tang, Jin Li, Xiao Xiao, Lin-Cheng Zhang, and Qing-Tian LV. Mathematical morphology filtering and noise suppression of magnetotelluric sounding data. *Chinese Journal of Geophysics*, 55(5):1784–1793, 2012.

Xinggang Wang, Wei Yang, Jeffrey Weinreb, Juan Han, Qiubai Li, Xiangchuang Kong, Yongluan Yan, Zan Ke, Bo Luo, Tao Liu, et al. Searching for prostate cancer by fully automated magnetic resonance imaging classification: deep learning versus non-deep learning. *Scientific reports*, 7 (1):1–8, 2017.

Yu Wang, Xiaojuan Ban, Jie Chen, Bo Hu, and Xing Yang. License plate recognition based on sift feature. *Optik*, 126(21):2895–2901, 2015.

Lintao Zheng and Guiping Qian. A sift-based approach for image registration. In *Green Communications and Networks*, pp. 277–287. Springer, 2012.

Huiyu Zhou, Yuan Yuan, and Chunmei Shi. Object tracking using sift features and mean shift. *Computer vision and image understanding*, 113(3):345–352, 2009.