Classification of date fruits in a controlled environment using Convolutional Neural Networks

Wadha S. N. Alhamdan and Jacob M. Howe

Department of Computer Science
City, University of London
London, United Kingdom
wadha.alhamdan@city.ac.uk, j.m.howe@city.ac.uk

Abstract. This paper explores the use of Convolutional Neural Networks in classifying images of date fruits as one of 9 varieties, creating several models with the highest achieving 97% accuracy. It contributes an original dataset of 1658 high-quality images taken in a controlled environment for use in both the computer vision and agricultural technology fields. A range of models is explored and trained, both with and without data augmentation, leading to high classification accuracy.

Keywords: Convolutional Neural Networks, supervised learning, classification, date fruit.

1 Introduction

Date fruits are a commodity for many Middle Eastern and African countries and hold both a cultural and religious importance. In 2017 the worldwide production of dates reached 8 million tons and the export value was at 1.63 billion US dollars in 2018 and is steadily increasing[15]. A mature palm tree, age 13 till around 60 years, will yield over 60 kg of dates per year, that is around 4.5-7 tons per acre [11]. Large amounts of dates need to be sorted based on type and grade so that they can be processed and priced accordingly. Many date types have similar physical properties and slight differences in colour, shape, fleshiness, etc. It is difficult to tell the different types apart, as there are many features to consider. Moreover, sorting through such large amounts is very time consuming and requires a lot of experienced and knowledgeable labour.

Whilst a number of papers discuss ways to automate the sorting and grading of fruit and vegetables with the help of machine learning, for example [2,3], there is little work that focuses on date fruits. Classification of dates is a particularly interesting machine learning problem since there are a large number of varieties of date, and as noted above many of these are similar in appearance. Currently, there are not many date fruit datasets available to work with. This work contributes to the computer vision and agricultural technology fields by providing a new image dataset in the area, and exploring machine learning for classification

of these images. This then connects the gap between newer technologies and date farms. By modernizing date farms, harvest processing times can be reduced and a more consistent sorting and grading can be achieved. The work can be built on to create a more extended date classification app which might be useful for consumers and authorities to check the labelling of products.

Convolution Neural Networks (CNNs) [10] are the current state-of-the-art for image classification and this paper explores the use of this deep learning technique to classify images of date fruits to their variety with high accuracy. The original dataset used consists of high-quality images taken in a controlled setting. A range of model architectures and the use of data augmentation are investigated in order to achieve high classification accuracy. The work has been setup so that it can be easily used by anyone and be built on or extended, for example by adding further varieties of date fruit, or incorporating the trained classifiers in a video stream. The following lists this paper's contributions:

- An original dataset that contains 1658 high-quality images of 9 types of dates taken under a controlled environment, which has been made available on Kaggle Kaggle link
- A set of successful model architectures is given, and the use of data augmentation and noise in their training is considered
- An empirical evaluation of the models, with the best trained model achieving a classification accuracy of 97% on the hold out test set.

2 Background

2.1 Neural Networks

Neural Networks (NN) consists of neurons connected by links weighted by real numbers, typically organised in layers, with the output of one layer feeding forward into the next. Each neuron calculates the product of its inputs and its weight set, and passes it into an activation function which determines an output (Rectified Linear Units are a popular choice of activation function with CNNs). Neurons reside in three different layer types: input, hidden, and output layers. For multi-class classification the activation function of the output layer is a Softmax function which outputs probability scores that are easier to interpret than raw data. Neural networks are typically trained using backpropagation, where the error determined by training data is used to update the weights in the network, typically using gradient descent [10,6].

2.2 Deep Learning

Deep learning refers to a wide variety of NNs with multiple hidden layers. It has made great strides in all the fields that have used it, whether in object detection or classification, speech recognition or in many other domains. Deep learning systems can be fed raw data and extract their own representation of it using different levels of abstraction, unlike in conventional machine learning

systems where representations are hard coded, making them limited [10]. A machine learning model goes through two phases: a training phase where the model learns from the data, and a testing phase where the model is tested on never seen before data (test data) and its predictions are compared against the true labels. Supervised learning is where the output of the function is known, allowing dynamic correction of a model's predictions whilst training.

2.3 Convolutional Neural Network Architecture

CNNs are a type of deep neural network that excels in processing images. They consists of a convolutional network connected to a neural network [6]. The convolutional network consists of convolutional layers with a number of filters and varying filter sizes. The filters perform a mathematical operation called convolutional which allows efficient extraction of different representations of images. These are then put through a feed forward neural network. Both sections have weights that are trained to extract representations and make predictions.

Although there is no fit-all model, it is useful to read and learn from others' experience. Common patterns in well-known models such as VGG16 [14] and Alexnet [9] inform this work, if on a smaller scale. Hyperparameters are the values that define the network structure and the learning process; they play a critical role in the model's performance. The goal was to create a relatively simple architecture that can achieve an over 90% accuracy (where accuracy is the percentage of correct predictions). The following is an overview of the hyperparameters:

- Number of layers
- Number of neurons
- Filter sizes and number
- Max pooling layers
- Dropout layers and L2 regularization
- Epochs, batch size and picture sizes
- Optimizers and learning rates
- Augmentation settings and noise layer

The number of layers and the number of neurons affect learning; more than two layers allows the model to learn complex representations, but too many can make the model overfit, and becomes slow to train. Filter number dictates how many representations to extract from the images. Smaller filter sizes extract smaller detail, larger sizes extract bigger structures in the images. Max pooling calculates the maximum value in each patch on the feature maps, producing pooled feature maps that are smaller. This work uses a patch size of 2x2, such that each dimension in the feature maps are halved. This is useful in generalizing the model and reducing amount of memory needed for training.

Dropout layers and L2 regularization are methods used to reduce and avoid overfitting in models. Dropout works by dropping out a percentage of randomly selected neurons or feature maps. L2 regularization works by making sure strong weight features do not overshadow weaker weight features. This is done by adding a penalty equal to the squared magnitude of a feature weight. In other words, the higher L2 is the more generalized the models are. Epochs determines the length of the training period; in each epoch batches are trained, and the larger

the batch size the faster the training goes, with a more varied batch passed through. Picture size is the input picture dimension passed into the model.

The work used several different optimizers: Adam [8], Adagrad [5], RMSprop [7], SGD. An optimizer updates the weights to minimize the loss from the loss function, the aim is to learn fast, not get stuck in a local minimum and not overlearn the training set. The learning rate dictates the size of the steps to make toward finding the global minimum in terms of loss.

Augmentation is a powerful tool that allows training sets to be made much larger and varied by making small changes to the original pictures. The following positional augmentations are tuned and applied: rotation range, vertical and horizontal shift range. Rotation range takes an integer for the degree range of random rotations. The shift range takes a float for the fraction of the total width or height of an image to pixel shift the image to. Noise helps mitigate overfitting by adding random data augmentation; the noise value tuned is a float that represents a standard deviation of the noise distribution.

3 Related Work

The subject of sorting fruit and vegetables is an area that has been bolstered by advances in computer vision using CNNs. Previous research would segment images then try to extract as many features as possible, often using a small dataset. For example, [13] considered potato chips set in different boiling temperatures and classified if they were blanched or not. 1511 features were extracted from 60 images, then 11 features chosen for classification using a decision tree. A similar paper collected images of various fruits and vegetables using intricate imaging systems [4]. In both papers, a large amount of work has gone into feature extraction, image acquisition and pre-processing. Computer vision papers before the boom in popularity of CNNs would often perform these procedures of manually extracting features and feeding them into different classifiers. CNNs can extract these representations automatically by learning filters, although this comes at the risk of not clearly seeing or understanding the features that are being used.

In [1], CNNs are used to classify five levels of growth of dates, providing a framework for a robot in an orchard environment that can quickly decide if they were ready for harvest or not. This paper used more than 8000 images, however, the images were not suitable for single date classification in a controlled environment. Another related paper collected over 1300 images of 4 different growth stages [12]. It used a CNN model to classify the 4 stages, including a defective stage. However, the growth stages were different enough in colour alone to tell them apart, unlike in this work where many date types are very close in colour or shape. The images in both works were not in a controlled environment, in terms of date position, camera focus, angle of capture, lighting conditions, and camera distance. In conclusion, computer vision topics relating to dates are under-researched which is the motivation for this work. Providing well documented work and accessible data can increase interest in this research area.

4 Methodology

4.1 Dataset Creation

Nine different types of dates were considered: Ajwa, Sokari, Galaxy (subtype of Sokari), Medjool, Meneifi, Nabtat Ali, Rutab (growth stage), Shaishe, Sugaey. A controlled environment was constructed to take pictures of the 9 different types. The imaging setup consists of a mounted DSLR camera (Canon EOS 550D) with the flash enabled, a ring light with a 48 centimetre diameter, and 240 led bulbs set to 100% brightness. A ring was used to negate any shadows by surrounding the date with light on all sides, the flash on the camera provides a strong sudden light to the centre, to emphasize the fleshiness or flabbiness of the date. The dates were put on an elevated platform on a white background. The distance between the background and the camera was maintained for all pictures which was 8 cm. The zoom and focus were maintained for all pictures. A large dataset was constructed, instances of each date type were put into its own folder and named according to type.

4.2 Pre-processing the dataset

Each image's 0-255 RGB values are mapped to the 0-1 interval, as is often used with NNs to make the computations faster and more precise. The high resolution images need to be resized to suitable dimensions for CNNs to work with; after testing, 120x120 was settled on. No form of object detection was used nor were the images cropped. All the recorded models have the same split percentages. The dataset is first split into training set and (hold out) testing set 70%-30%, then 20% of the training data is taken for the validation set and provided to the fit method (that trains the model). After each training epoch the model is tested against a batch of the training and validation set, with the loss and accuracy on these sets saved to create plots. In other words, the dataset is split 56%-14%-30% for training, validation, testing sets, respectively.

4.3 Model Development

The work was coded in Python 3.7.5, using TensorFlow 2.0, keras 2.2.4 and minor use of other libraries. It explored a wide variety of models to achieve the highest accuracy possible on the test set and create consistent models with little fluctuations in their plots. The recorded models are tagged as their chronological order followed by their convolutional layer number, such as 8-4L is the 8th model in the 4 convolutional layers model architecture. Four final models (48-4L, 65-4L, 70-4L, 74-4L) trained under different augmentation and noise settings were found, each excelling in its own category; these will be the focus of the paper.

The project was structured into 3 model development phases leading to each of the final 4 models with the last phase outputting 2 final models. Each phase focused on specific hyperparameters or added new features to the architecture. In each phase, a set of hyperparameters are chosen and investigated as such: a

commonly chosen value and position for the parameter is first tested, this value is incremented and decremented until the two edge values are found. Edge values are when the model starts to falter or when it has no effect on the model. Once a suitable value or range of values is found, different positions on the architecture are tested when applicable. In-between models are recorded to uphold the documented rationale for the next steps taken. For example, in the development leading to the 1st notable model, L2-regularization was added on every convolutional layer with a starting value of 0.005. This value was incremented/decremented by a factor of 10, edge values found were 0.000000005 where the L2 had no notable effect and 0.005 where the L2 was generalizing the model too much to the point of under performing. It was found that an L2 between [0.0005, 0.00005] is more suitable on the models with no augmentation or noise. Each model architecture was trained and tested multiple times to find the average performance.

In the 1st phase a 2-layer model was the first created and the first hyperparameter tuned was dropout. That model was extended to 4 convolutional layers, 3 feed forward layers and higher batch sizes, filter and neuron numbers model. The next hyperparameters to be tuned were the batch size and picture dimension, followed by the L2 value, then the number of filter and the number of neurons and their patterns. This leads to the first of the final models (48-4L). The filter and neuron patterns for model 48-4L (tuned for no augmentation) were inspired by popular CNN models [9,14]. Model 48-4L defined the basic architecture for later 4-layer models, which excels on the dataset as is.

The next phase adds data augmentation and tunes the non fixed parameters to find a model that works best on augmented data. Augmentation values were chosen by viewing the augmented pictures and considering the maximum displacement a date would be in a real life sorting machine. To better generalize and reduce overfitting, the batch size and L2 value were increased. These two hyperparameter values became fixed for future models. Lastly, different optimizers, learning rate values, and epochs were tested. The second notable model (65-4L) was created in this phase.

At this point noise layers were added, different noise values on different levels were tested and eventually a noise value of 0.02 was set at the input layer. Many hyperparameters are now fixed and only small adjustments in the number of epochs, optimizer type and learning were done. The last two final models, noise added, with and without augmentation, come from this phase: 70-4L and 74-4L.

5 Results

5.1 Dataset

The dataset was created using dates from Saudi Arabia and contains types that are not easily found elsewhere. Similar-looking types were chosen, to make the problem of classifying them hard enough to pursue solving. A total of 1658 high quality pictures with the following number for each type was collected: Ajwa=175, Galaxy=190, Medjool=135, Meneifi=232, Nabtat Ali=177,

Fig. 1. Sample from dataset, resized and cropped evenly from the sides

Rutab=146, Shaishe=171, Sokari=264, Sugaey=168. The dataset of high resolution images is 3.14Gb large and 3.11Gb compressed, and is available on Kaggle, Kaggle link. Figure 1 gives sample images of each date type. Image sizes were fixed to 120 x 120 for use in training and testing.

5.2 Models' Performance

This section will describe the results that came from the development phases, focusing on the 4 notable models. In general, it was found that 2-layer models (2 convolutional layers) were too small to experiment with and that 8-layer models took too much time training while not significantly surpassing the 4-layer models. Table 1 gives accuracy on the hold out test set for notable models under different treatments of the data.

Table 1. Accuracy of the best performing models in different categories

Category	Best performing model
No augmentation, no noise	48-4L with 96%
With augmentation, no noise	65-4L with 95%
No augmentation, with noise	70-4L with 97%
With augmentation, with noise	74-4L with 91%

The first model created was 1-2L, it had 2 convolutional layers with the first having 32 filters of size 3x3 and the second 64 of size 5x5. Moreover, it had 2 feed forward neural layers with 128 and 9 neurons, note that the output layer neurons must match the number of classes the model need to classify. Max pooling was always applied at the end of both convolutional layers. This model used a very strong dropout configuration of 80% on the input layer. This first model provides a baseline result for the rest of the experimental work and resulted in an accuracy of 65% on the test set.

Model 48-4L defined the core architecture for later models, that is, some important hyperparameters were now fixed while tweaking others. The bold hyperparameters in Table 2 are the fixed ones. 48-4L has pooling after the 1st, 2nd, and 4th convolutional layer but not after the 3rd (as seen in Figure 2), this is to increase the amount to be learned. There are dropout layers after each of the 3 feed forward layers but not on the classification layer. The dropout value

Table 2. 48-4L Architecture

Model name:	48-4L	Optimizer and lr:	SGD, lr = 0.001
Batch size:	32	**Filter numbers:**	**(96, 128, 256, 256)**
Picture size:	**120x120**	**Filter sizes:**	**(5x5, 5x5, 3x3, 3x3)**
Epochs:	200	**Neurons:**	**(1024, 1024, 1024, 9)**
L2 value:	0.0005	**Max pooling (2x2)**	**(y, y, n, y)**
Accuracy:	95%	**Dropout on FC:**	**(50%, 50%, 50%, 0%)**

in the feed forward layers is 50%. Its classification report shows an average of 96% f1-score (weighted average of the precision and recall), highest for Ajwa with 100% and lowest for Meneifi with 93% (see the confusion matrix in Figure 3). It has textbook-like loss and accuracy plots where the validation line sticks closely to the training with very little fluctuation, as seen in Figure 3.

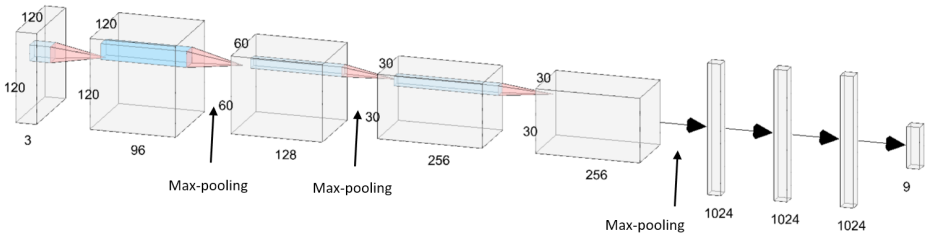

Fig. 2. Illustrative diagram of the architectire of the final models

The next notable model is 65-4L which uses Adagrad with default learning rate of 0.01 and produces 95% accuracy. This is the highest accuracy a model produced with the following augmentation [rotation = 20, shift = 0.2] and no noise. In this model the L2 has increased to 0.005, batch size to 50, and epoch to 1000. 65-4L's accuracy plot fluctuates between the range of [50%, 85%] for the validation set and its loss validation line closely follows the train set loss.

The final phase experimented with adding noise layers. Model 70-4L uses noise and Adam as the optimizer with learning rate of 0.000001 and epoch=1500, trained without augmentation. Multiple learning rates were tested to find the most suitable one. Model 70-4L achieves a very high accuracy of 97%, slightly higher than the SGD model without augmentation (model 48-4L with accuracy of 96%). Like 48-4L, the validation line sticks very close to the training line in its loss and accuracy plots. The last notable model is 74-4L which uses both augmentation and noise, with Adagrad as the optimizer. This produces 91% test set accuracy. The 74-L plots (Figure 3) resemble the 65-4L plots but are slightly less noisy near the end but not very stable.

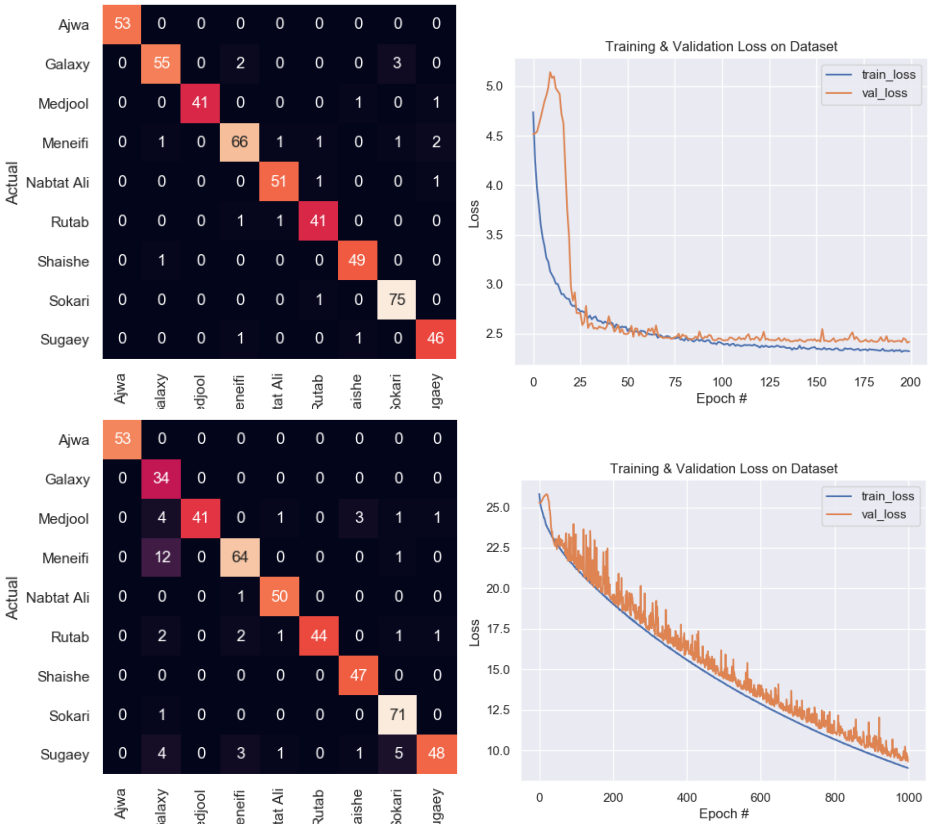

Fig. 3. Models 48-4L (top) and 74-4L(bottom) confusion matrix (left) and loss plots (right)

6 Conclusion

The primary objective of this work was to create an original dataset of date fruit images and find a model with over 90% accuracy. This goal was achieved, finding several models with over 90% accuracy under different treatments of the input data. This leads to the conclusion that there are a range of models able to achieve high accuracy and that it is almost never a one model fits-all situation. Instead, one should look to popular models for inspiration and adapt as needed, as seen in [1] where they tested different popular models against their own created models which performed slightly better and faster.

The notable models were incorporated into a video setup to provide an initial investigation into their classification in real-time. The video setup does not follow the controlled environment of the training dataset, in particular lighting and brightness vary. This leads to weaker results than on the controlled environment dataset, and motivates future work on images from less controlled environments.

The original dataset produced contributes directly to the machine learning and agriculture technology fields by providing a high-quality dataset of a large selection of date types (including types not widely available) taken in a controlled environment. The dataset is important because there are not any date datasets available in the form of single date per image in a controlled environment. By providing an easily accessible dataset and a well documented set of models this work sets the challenge of trying to improve on the 97% accuracy of the best performing model, whilst also motivating further work on classification of similar looking images under varying lighting conditions.

References

1. Altaheri, H., Alsulaiman, M., Muhammad, G.: Date Fruit Classification for Robotic Harvesting in a Natural Environment Using Deep Learning. IEEE Access **7**, 117,115–117,133 (2019). DOI 10.1109/access.2019.2936536
2. Anurekha, D., Sankaran, R.A.: Efficient classification and grading of MANGOES with GANFIS for improved performance. Multimedia Tools and Applications **79**(5-6), 4169–4184 (2020)
3. Bhatt, A.K., Pant, D.: Automatic apple grading model development based on back propagation neural network and machine vision, and its performance evaluation. AI and Society **30**(1), 45–56 (2013)
4. Cubero, S., Aleixos, N., Moltó, E., Gómez-Sanchis, J., Blasco, J.: Advances in Machine Vision Applications for Automatic Inspection and Quality Evaluation of Fruits and Vegetables. Food and Bioprocess Technology **4**(4), 487–504 (2011)
5. Duchi, J., Hazan, E., Singer, Y.: Adaptive subgradient methods for online learning and stochastic optimization. Journal of Machine Learning Research **12**, 2121–2159 (2011)
6. Goodfellow, I., Bengio, Y., Courville, A.: Deep Learning. MIT Press (2016)
7. Hinton, G.: Unpublished adaptive learning rate method proposed in the author's Coursera Class (6e), URL http://www.cs.toronto.edu/~tijmen/csc321/slides/lecture_slides_lec6.pdf
8. Kingma, D.P., Ba, J.L.: Adam: A method for stochastic optimization. In: 3rd International Conference on Learning Representations, ICLR 2015 - Conference Track Proceedings (2015)
9. Krizhevsky, A., Sutskever, I., Hinton, G.E.: ImageNet classification with deep convolutional neural networks. Communications of the ACM **60**(6), 84–90 (2017)
10. LeCun, Y., Bengio, Y., Hinton, G.: Deep learning. Nature **521**(7553), 436–444 (2015). DOI 10.1038/nature14539
11. Morton, J.: Fruits of Warm Climates, chap. Date, pp. 5–11 (1987)
12. Nasiri, A., Taheri-Garavand, A., Zhang, Y.D.: Image-based deep learning automated sorting of date fruit. Postharvest Biology and Technology **153**(April), 133–141 (2019). DOI 10.1016/j.postharvbio.2019.04.003
13. Pedreschi, F., Mery, D., Mendoza, F., Aguilera, J.M.: Classification of potato chips using pattern recognition. Journal of Food Science **69**(6), 264–270 (2004)
14. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. In: 3rd International Conference on Learning Representations, ICLR 2015 - Conference Track Proceedings (2015)
15. Tridge: overview of global date market (2019). URL https://www.tridge.com/intelligences/dates. Accessed 03/10/2019