On the usage of Neural Networks for Image Classification

Pedro N. Mota
Student, Graduate Researcher
Department of Computer Science
University of Porto
Email: up201805248@up.pt

Rúben A. Dhanaraju
Student, Graduate Researcher
Department of Computer Science
University of Porto
Email: up201706353@up.pt

In this paper, we study how the in state-of-the-art computational models recognize images. Our focus will be neural networks, in particular, deep neural networks. In contrast to the human recognition model, deep neural network models require a large number of training instances in order to distinguish a cat from a dog, and if the images are noisy, the models lose their predictive power. Our main objective is to understand why deep learning alone may not be a good learning model and explain how we could achieve a good computational human-like model.

1 Introduction

Our visual system and brain work together in order to identify and recognize images and objects with a very high speed and accuracy, as long as they are not impaired. A 2-year old child needs to see and be introduced to a small number of images and explanations in order to distinguish, say, a cat from a dog or any other animal, even if the image is blurred or the object is partially blocked. The state-of-the-art computational model to recognize images is based on neural networks, in particular, deep neural networks composed by a number of layers. In contrast to the human recognition model, deep neural network models require a large number of training instances in order to distinguish a cat from a dog, and if the images are noisy, they lose their predictive power. Other problems arise when new classes are added, that is, we have now to distinguish between cats, dogs, and monkeys and they if the observations are imbalanced. We start by describing the models which were used on this project, as well as the data that was used to access their performance. We present and analyse these results, and in further sections we introduce annotations to the training. We finally present a Reinforcement Learning approach to solve this classification problem, and subsequently end with a showcase of recent investments in combining logic with deep learning. We also propose a method which falls in this category.

2 Used Datasets

In this section, we present an overview of the datasets which were used to access the performance of our models. We prepare the data as in [1].

Dogs vs Cats Dataset
The dogs vs cats dataset was created for a Kaggle machine learning competition that took place in 2013. Pictures of dogs and cats are included in the collection, which is a subset of a much larger dataset of 3 million manually annotated photos. The dataset was created as a result of a collaboration between Petfinder.com and Microsoft.

Noisy Dogs vs Cats Dataset
In order to benchmark the performance of our models, we modified the original dataset with the adition of noise. The noise that was added is called *Gaussian white* noise, *Gaussian* due to having a normal distribution in the time domain, with an average time domain value of zero. It is also denominated *White* because it has uniform power across the frequency band for the information system, like the color white which has uniform emissions at all frequencies in the visible spectrum.

For this project, the noise was added with a variance of 0.05, to about 30% of our data set. In further sections, we present the results we obtain for the VGG arc, with one, two and three blocks, respectively.

Figure 1 shows, an image of a dog before and after the add of noise, left-to-right, respectively.

Dogs vs Cats vs Birds

Besides adding noise, in order to measure the performance of the models, we also introduce the birds class. This consists of images taken from [2] with the same preprocessing steps of [1].

3 Used Models

In this section, we describe the models which were developed and trained for this paper.

Fig. 1

Baseline Model

In this section, we describe the baseline convolutional neural network model to identify cats vs dogs. This model will establish a minimum model performance to which all of our other models can be compared, as well as a model architecture that we can use as the basis of study and improvement.

A good starting point is the general architectural principles of the VGG models. These are a good starting point because they achieved top performance in the ILSVRC 2014 competition and because the modular structure of the architecture is easy to understand and implement. For more details on the VGG model, the reader may refer to [3].

The architecture involves stacking convolutional layers with small 3×3 filters followed by a max pooling layer. Together, these layers form a block, and these blocks can be repeated where the number of filters in each block is increased with the depth of the network such as 32, 64, 128, 256 for the first four blocks of the model. Padding is used on the convolutional layers to ensure the height and width shapes of the output feature maps matches the inputs.

We can explore this architecture on the dogs vs cats problem and compare a model with this architecture with 1, 2, and 3 blocks.

Each layer will use the ReLU activation function and the He weight initialization, which are generally best practices.

The models will be fit with stochastic gradient descent and we will start with a conservative learning rate of 0.001 and a momentum of 0.9.

We fixed the batch size at 64 and fitted the model for 20 epochs.

The problem is a binary classification task, requiring the prediction of one value of either 0 or 1. An output layer with 1 node and a sigmoid activation will be used and the model will be optimized using the binary cross-entropy loss function.

Dropout Regularization

In order to avoid overfitting, we extend the three-block VGG model described previously. Dropout regularization [4, 5] is a low-cost method of regularizing a deep neural network.

Dropout works by probabilistically deleting, or "dropping out," inputs to a layer, which could be data sample input variables or prior layer activations. It has the effect of simulating a wide variety of networks with extremely various network architectures, making network nodes more resilient to inputs in general.

Image Data Augmentation

Image data augmentation [6, 7] is a technique for increasing the size of a training dataset artificially by producing changed versions of the images in the dataset. Data augmentation can also be used as a regularization strategy, by introducing noise to the training data and encouraging the model to learn the same features regardless of where they appear in the input. Deep learning neural network models that are trained on more data become more skilled, and augmentation approaches can provide variations of the images that increase the fit models' capacity to generalize what they've learned to new images.

Transfer Learning with VGG16

K. Simonyan and A. Zisserman of the University of Oxford proposed the VGG16 convolutional neural network model in [8].In ImageNet, a dataset of over 14 million images belonging to 1000 classes, the model achieves 92.7 percent top-5 test accuracy. It was a well-known model that was submitted to the ILSVRC-2014.It outperforms AlexNet by sequentially replacing big kernel-size filters (11 and 5 in the first and second convolutional layers, respectively) with numerous 33 kernel-size filters.

Transfer Learning with ResNet

Residual Networks, or ResNets, proposed by He et. al. in [9] instead of learning unreferenced functions, learn residual functions with reference to the layer inputs. Residual nets allow these layers to suit a residual mapping rather than expecting that each few stacked layers directly match a desired underlying mapping. They build networks by stacking residual blocks on top of each other: a ResNet-50, for example, has fifty layers made up of these pieces. There is evidence that these types of networks are easier to optimize and that they can gain accuracyfrom significantly higher depth.

Transfer Learning with Xception

Xception is a Depthwise Separable Convolutions-based deep convolutional neural network architecture, proposed by Google researchers in [10]. Inception modules in convolutional neural networks are described by Google as an intermediate step between normal convolution and the depthwise separable convolution operation (a depthwise convolution followed by a pointwise convolution). In this sense, a depthwise separable convolution can be thought of as an Inception module with the most towers possible. This result leads them to propose a new deep convolutional neural network architecture based on Inception, but with depthwise separable convolutions in place of Inception modules.

4 Experimental Results

In this section, we access the performance of the previously described models on the datasets presented in section 2. Table 2 depicts the accuracies for the unaltered Cats vs Dogs dataset, using the models from [1] plus the VGG16, ResNet and Xception models. Table 3 shows a comparison of the accuracies of all models except the latter, for the different datasets.

Looking first at Table 2, we establish the baseline performance as roughly 73%. As we increase the blocks on the VGG, the accuracy tends to increase, reaching about 80%. Dropout regularization does not increase substantially the accuracy, as it remains in 80%, however, the application of Data regularization increases it largely to 86%. With regards to the transfer learning models, starting with the VGG16 we dramatically increase the accuracy to almost 98%. The ResNet model obtains approximately 98.6%, however Xception achieves 73.425% beating the baseline model by roughly 0.2%.

Analysing Table 3, we notice that the introduction of noise or another class decrease the accuracy of the models. With regards to the noisy dataset, we verify that its accuracy for the models ranges from slightly above 50% to a bit down from 60%. When introducing another class, all models achieved an accuracy of about 67%. Looking at the results, the worst model was the baseline and the best one was the model where Data Augmentation was introduced.

5 Using annotations

We will now try another way of training by using a description of the images.

The idea comes from the fact that us, humans, do not just look at an image as simply a collection of many small square pixels. We pick out details, or features, from images in order to identify what it is you are looking at.

Accordingly to this, we choose a set of features that we considered that would be useful to distinguish dogs and cats. After a careful consideration, we choose the following features:

1. *homogeneous_body_color*: a binary variable, having the value 1 if the animal as an homogeneous body color, 0 otherwise.
2. *body_size*: an ordinal variable, taking the values *small*, *medium*, *large*, referring to the size of the animal
3. *eye_color*: a nominal variable, taking the values *yellow*, *green*, *black*, *brown* and *blue*, referring to the animal's eye color.
4. *nose_size*: an ordinal variable, taking the values *small*, *medium*, *large*, referring to the size of the animal's nose
5. *ears_shape*: a nominal variable, taking the values *round* or *pointy*, referring to the shape of the ear
6. *animal*: the binary target variable, having the values *dog* or *cat*, in order to identify the animal

As said, the feature extraction was rather challenging, because they can be irrelevant, thus providing no useful information to the classifier.

For instance, regarding the distinction between cats and dogs, a feature like "number_of_legs", describing the number of legs of the animal, would be unhelpful in discriminating between cats and dogs, since they both have four.

Having deciding the set of features to be used, we labeled, manually, a set of 62 images of cats and dogs, that we also chose carefully, in order to be easy to properly extract the chosen features and, also, to have a good variance of pictures, since, for instance, most of the cats have small noses and pointy ears, while most of the dogs have big noses and round ears.

Since we have now tabular data, we decide to use a Random Forest, with 500 trees, and split at each node to a subset at most of 3 variables, as our classifier. For our labeled data set, we obtained an accuracy of about 71%.

During this process, we noticed that the feature *eye_color* was, by itself, describing almost, perfectly, whether if the animal was a dog or a cat. This is because we had a small data set, so we decided to drop this feature, in order to make the results more interesting.

5.1 With an imbalanced data set

We shall try now, with the same setup, but having less, say, dog observations. Our objective is to study how a imbalanced data set may influence our final classification model.

We trained our Random Forest with a number of cats equal to the double of the number of dogs. In this case, we obtained an accuracy of 62%. What we noticed is that the model predicts a lot of cats as dogs, which results in a lower accuracy.

We also tried with a data set even more imbalanced and what we found is that the accuracy decreased even more.

5.2 With a new animal

We will now add a new class of animal and see how the difference in our models performance.

The animal that we choose is a bird and we added a new feature to our data set:

1. *wings*, a binary variable, having the value 1 if the animal has wings or 0, otherwise

What we found is that our classification model improved, a lot, because it always had a recall and precision of 1 regarding the birds, since they can be easily identified by the presence of wings, which neither the dogs or cats have.

But we saw, also, for reading the literature, is that this isn't, normally, the case, that is, the model's performance tends to decrease as new classes are added and this decrease is more significant as the new animal is similar to the already present. The reason here was simply because the new added animal was significantly morphologically different from the already existing animals.

5.3 Combining the annotations with the images

We describe an approach that combines both the images of the animals and the annotations presented in the previous

Model	Accuracy (%)
Baseline	73.187
1B VGG	73.600
2B VGG	74.869
3B VGG	80.216
D. Reg.	80.010
D. Aug.	86.038
VGG16	97.890
ResNet	98.572
Xception	73.425

Fig. 2: Accuracies (%) for models on the original Cats vs Dogs dataset

Dataset	Baseline	1B VGG	2B VGG	3B VGG	D. Reg.	D. Aug.
Dogs vs Cats	73.187	73.600	74.869	80.216	80.010	86.038
Dogs vs Cats with Gaussian White noise	52.640	54.410	55.721	56.462	56.962	57.628
Dogs vs Cats vs Birds	66.667	66.667	66.667	66.667	66.667	66.667

Fig. 3: Accuracies (%) for models on different datasets

chapter. First, the annotations are used to predict the probabilities of being a dog or a cat, with a model such as Random Forest. Subsequently, we use a neural model that uses the images to achieve the same thing. These probabilities are them summed, and we take the maximum to perform classification.

In this case, what we found is that the combination from annotations and the images improve our performance in the classification task, a lot. Using a two block VGG for the neural network to classify the images and a Random Forest as a model to the tabular data, we obtained an accuracy of about 88% percent.

6 Reinforcement Learning approach

We propose a hybrid approach of deep learning and Reinforcement Learning, based on [11]. We use a ResNet50 CNN [9, 12, 13]. First, we train the network using the training data as usual. We subsequently classify the testing set using the same CNN, however we try to increase the accuracy with the aid of Reinforcement Learning, more specifically, Q-learning [14–16], with a random policy. In short, for the Q-learning there is a set of actions which are defined as transformations to the images and two states (before the transformation, and after). For this experiment, we defined four different actions, each of which rotates the image by a specific angle.

The reward is defined formally as follows. We first define metrics M_0 and M_a, where M_0 is the standard deviation of the prediction scores of the CNN before the Q-Learning process and M_a after the application of action a on the image. The reward r is determined as shown bellow:

$$r = \begin{cases} +1 & \text{if } M_0 > M_a \\ 0 & \text{if } M_0 = M_a \\ -1 & \text{if } M_0 < M_a \end{cases}$$

The Q-Table will have, by our previous definition, two rows and four columns, with all cells initialized to zero. We update the Q-table iteratively, using the Q-Learning update rule:

$$Q(s,a) = Q(s,a) + \alpha \times [r + \gamma \times \max_{\forall b \in A} Q(s',b) - Q(s,a)]$$

Where s is a state, and a is an action, s' is the new state after applying action a. We set the learning rate to $\alpha = 0.5$ and the discount rate $\gamma = 0.3$. After the maximum number of iterations, we choose the action with the highest Q-Value. We finally apply this optimal action to the test image, and feed it to the CNN and classify it.

7 Logic thinking in Deep Learning

In 2019, DeepMind tried to answer the question of whether deep learning could achieve logic thinking in [17]. They implemented a transformer model to solve mathematical problems. This model reached above 90 % accuracy in simple addition, subtraction, division and multiplication, however it drastically dropped when mixing these operations. This suggests that the model was simply guessing the solution, instead of solving it.There are additional cases in deep learning when the models or agents are so good at their

jobs that they create logical and reasoning illusions. Integration of deep learning with logic thinking is still a open research problem, but it is seen as critical to the development of true intelligent agents. Marra et. al present in [18] Deep Logic Models, which consist of deep graphical models integrating deep learning and logic reasoning for both learning and inference. The ability to fully merge learning from low-level representations with semantic high-level reasoning over network outputs is the fundamental benefit of the described system. Allowing the weights of deep learners and the parameters regulating reasoning to be learned together allows for a positive feedback loop, which has been demonstrated to enhance the accuracy of both layers.

Another approach was the development of a fundamentally new neuro-symbolic thecnique called Logical Neural Networks (LNN), where artificial neurons comprise a notion of weighted real-valued logic [19]. LNNs inherit essential aspects of both neural networks and symbolic logic by design and can be used to reason with domain knowledge.

By using a recursive neural calculation of truth values that flows both forward and backward, LNNs can mimic formal logical reasoning (whereas a standard neural network only moves forward). As a result, LNNs are more understandable, capable of tolerating imperfect knowledge, and capable of full logical expressivity. This networks have shown very promising experimental results [20].

We propose an unimplemented approach to discuss how to take a step further into achieving the holy grail of Artificial Intelligence. When we consider how we learn what a cheetah is, for example, we recognize that is a big feline with spots. A lion, on the other hand, is a big feline with no spots and with a mane. One might consider the task of extracting high level features from what we are classifying and train machine learning models, such as Convolutional Neural Networks, to identify these features. By using a decision tree, or logic statements, combined with a CNN, we could theoretically identify the animals in a quicker way, without submitting these models to intensive training.

8 Conclusions

Deep Learning has been rising in popularity over the last years, however, it is evident that it is not perfect. Even though these models may classify some datasets with very good accuracy, the introduction of obstacles such as noise will decrease its performance. The same happens when introducing other classes to the dataset.

We also show that by training with an unbalanced dataset, we are able to decrease the accuracy of the models. In contrast, a Human would not need the requirement of having a balanced dataset, because Humans use logical thinking and abstract reasoning. While the models use the pixel values to extract features, they model a mathematical function to represent the data, as opposed to generalizing and learning concepts.

As mentioned in the previous section, there are novel models which try to tackle this problem by combining logic with deep learning, however this is still an open research problem in Artificial Intelligence.

Finally, we proposed a hybrid Deep Learning and Reinforcement Learning approach for this problem, which tries to increase the accuracy of the classifications using Q-Learning, which we were not able to correctly test.

References

[1] Brownlee, J., 2021. How to classify photos of dogs and cats (with 97% accuracy), Dec.

[2] Gerry, 2021. 325 bird species - classification, Dec.

[3] Simonyan, K., and Zisserman, A., 2015. Very deep convolutional networks for large-scale image recognition.

[4] Wager, S., Wang, S., and Liang, P. S., 2013. "Dropout training as adaptive regularization". *Advances in neural information processing systems,* **26**, pp. 351–359.

[5] Jindal, I., Nokleby, M., and Chen, X., 2016. "Learning deep networks from noisy labels with dropout regularization". In 2016 IEEE 16th International Conference on Data Mining (ICDM), IEEE, pp. 967–972.

[6] Shorten, C., and Khoshgoftaar, T. M., 2019. "A survey on image data augmentation for deep learning". *Journal of Big Data,* **6**(1), pp. 1–48.

[7] Perez, L., and Wang, J., 2017. "The effectiveness of data augmentation in image classification using deep learning". *arXiv preprint arXiv:1712.04621*.

[8] Simonyan, K., and Zisserman, A., 2014. "Very deep convolutional networks for large-scale image recognition". *arXiv preprint arXiv:1409.1556*.

[9] He, K., Zhang, X., Ren, S., and Sun, J., 2016. "Deep residual learning for image recognition". In Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 770–778.

[10] Chollet, F., 2017. "Xception: Deep learning with depthwise separable convolutions". In Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 1251–1258.

[11] Hafiz, A. M., 2020. "Image classification by reinforcement learning with two-state q-learning". *arXiv preprint arXiv:2007.01298*.

[12] Xie, S., Girshick, R., Dollár, P., Tu, Z., and He, K., 2017. "Aggregated residual transformations for deep neural networks". In Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 1492–1500.

[13] He, K., Zhang, X., Ren, S., and Sun, J., 2016. "Identity mappings in deep residual networks". In European conference on computer vision, Springer, pp. 630–645.

[14] Jang, B., Kim, M., Harerimana, G., and Kim, J. W., 2019. "Q-learning algorithms: A comprehensive classification and applications". *IEEE Access,* **7**, pp. 133653–133667.

[15] Watkins, C. J. C. H., 1989. "Learning from delayed rewards".

[16] Watkins, C. J., and Dayan, P., 1992. "\cal q-learning". *Machine Learning,* **8**(3-4), pp. 279–292.

[17] Saxton, D., Grefenstette, E., Hill, F., and Kohli, P.,

2019. “Analysing mathematical reasoning abilities of neural models”. *arXiv preprint arXiv:1904.01557*.

[18] Marra, G., Giannini, F., Diligenti, M., and Gori, M., 2019. “Integrating learning and reasoning with deep logic models”. *arXiv preprint arXiv:1901.04195*.

[19] Riegel, R., Gray, A., Luus, F., Khan, N., Makondo, N., Akhalwaya, I. Y., Qian, H., Fagin, R., Barahona, F., Sharma, U., et al., 2020. “Logical neural networks”. *arXiv preprint arXiv:2006.13155*.

[20] Trivedi, P., Maheshwari, G., Dubey, M., and Lehmann, J., 2017. “Lc-quad: A corpus for complex question answering over knowledge graphs”. In International Semantic Web Conference, Springer, pp. 210–218.