Evaluation of Transfer Learning Techniques in Neural Networks with Tiny-scale Training Data

Gabriela Pérez[1,2], Milagros Jacinto[1], Martín Moschettoni[1], Claudia Pons[1,3,4]

[1] Facultad de Informática, Universidad Nacional de La Plata, Argentina

[2] UNAJ, Universidad Nacional Arturo Jauretche, F. Varela, Buenos Aires, Argentina

[3] Comisión de Investigaciones Científicas CIC, Bueno Aires, Argentina

[4] UAI, Universidad Abierta Interamericana, Ciudad de Buenos Aires

{Gperez,Mjacinto,Mmoschettoni,Cpons}@lifia.info.unlp.edu.ar

Abstract. *This paper rigorously analyzes the process of building a deep neural network for image recognition and classification using Transfer Learning techniques. The biggest challenge is assuming that the training dataset is very small. The research is based on addressing a particular case study, the income of donations to the Food Bank of La Plata. The results obtained corroborate that the techniques analyzed are appropriate to solve tasks of detection and classification of images even in cases in which there is a very moderate number of samples.*

Resumo. *Este artigo analisa rigorosamente o processo de construção de uma rede neural profunda para reconhecimento e classificação de imagens usando técnicas de Transfer Learning. O maior desafio é assumir que o conjunto de dados de treinamento é muito pequeno. A pesquisa se baseia em abordar um estudo de caso particular, a receita de doações ao Banco de Alimentos de La Plata. Os resultados obtidos corroboram que as técnicas analisadas são adequadas para resolver tarefas de detecção e classificação de imagens mesmo em casos em que há um número muito moderado de amostras.*

Keywords: machine learning, transfer learning, pre-trained models, small dataset, food bank, Keras, Convolutional Neural Networks.

1 Introduction

Machine learning (Alpaydin, 2016) is an area within Artificial Intelligence (Russell & Norvig, 2021) whose objective is to develop techniques that allow computers to learn to perform specific tasks effectively. It focuses on building predictive mathematical models from large sample or training datasets. One of the most used methods applies Artificial Neural Networks (ANNs) (Goodfellow , Bengio, & Courville, 2016) which is a biologically inspired approach to machine learning, inspired by the functioning of the nervous system and learning in living organisms. Within this approach, the Deep Learning (DL) technique involves the construction of artificial neural networks with many layers that are iteratively trained using large data sets (LeCun, Bengio, & Hinton, 2015).

A vital aspect in capturing knowledge is the possibility of storing and reusing it, so that each new artificial intelligence system does not have to start learning from zero. The technique that addresses this problem is called Transfer Learning (Jialin & Yang, 2010) and basically consists of building models trained to large scale, for the purpose of being reused. These models are called pre-trained models (Han, 2021). AlexNet (Krizhevsky, Sutskever, & Hinton, 2017), BERT and GPT (Hugging-Face, s.f.) are good examples of this kind of models. They have achieved great success and have become a milestone in the field of artificial intelligence. These models have tens of millions of parameters and have been trained on big data. That way they effectively capture the knowledge and can be adjusted to solve similar problems. One of the advantages of working with the transfer learning technique is the possibility of achieving an effective model with little training data.

In order to analyze this characteristic, a case study was developed: the classification of products for the Food Bank of La Plata (http://bancoalimentario.org.ar). This organization aims to recover food, to be distributed to community organizations. It works with donations whose classification is a cumbersome task due to its diversity and quantity, which is currently carried out by manual means. The objective of this case study is to verify the feasibility of building a Machine Learning model to properly classify the food received in donations but counting with very few images for training.

This article is organized as follows. In Section 2, the Transfer Learning technique applied to image recognition and classification using artificial neural networks is described. The different training strategies and their development process are explained. In Section 3 the food bank of La Plata that will be used as a case study, is presented. Then, in section 4, each of the steps taken to build the image recognition model for food products is discussed. To this end, different models are trained and compared to analyze the factors that determine their effectiveness, considering that very little training data is available. Finally, section 5 presents the conclusions and lines of future work.

2 Transfer learning concepts

Transfer learning is an area within deep learning, which focuses on the incorporation of knowledge obtained during the resolution of a certain problem and its subsequent application on a different but related one (Jialin & Yang, 2010). Using the traditional Deep Learning technique, models must be trained from scratch. Instead, using Transfer Learning a knowledge base is built from the training of source tasks, so that it can be transferred to another target task, requiring less quantity and quality of training data (Torrey & Shavlik, 2009). In this case, the new model does not learn from scratch. Instead, a model previously trained to solve another task is used as a starting point.

There are three measures to know the improvements produced using Transfer Learning:

-The first is the initial performance that can be achieved on the target task, using only the transferred knowledge, before any further learning is undertaken.

-The second measure is the level of final performance achievable in the target task.

-And the third is the amount of time it takes to fully learn the target task given the transferred knowledge.

The application of transfer learning methodologies increases the degree of generalization of deep learning models, allowing the use of smaller data sets (tens or hundreds of instances) and greater simplicity (low dimensionality, few features). In addition, it saves learning time and reduces required computing resources by making the requirement for high-performance hardware more flexible. This allows good performance of models with fewer iterations for training. Finally, it allows more robust and widely applied representations (Torrey & Shavlik, 2009).

2.1 Transfer Learning Strategies

Considering the question "what to transfer?" four approaches emerge, whose treatment allows to evaluate the particular aspects of each problem and apply the most appropriate techniques (Neyshabur, 2020).

These approaches are:

- Instance transfer: Certain parts of the source domain data are considered useful in training on the target domain and can be reused.
- Feature representation transfer: the contribution is the discovery and learning of a good feature representation of the target domain to reduce differences between domains and the error of the generated model.
- Parameter transfer: It is assumed that the source and target tasks share parameters or previous distributions of model hyperparameters. The transferred knowledge is encoded in shared parameters.
- Relational knowledge transfer: It assumes the existence of relationships between the data of both domains, assuming that they are relational domains. These relationships between domains represent the knowledge to be transferred by building relational knowledge mappings.

Pre-trained neural networks are a transfer learning technique that is based on combining instance transfer, feature representation and parameter transfer. Using deep neural networks, the knowledge acquired by solving a source problem is stored in the architecture of the network and in the weights of the neural connections. These parameters and hyperparameters can be re-used and adjusted, making it easier to build a model for the target problem.

2.2 Transfer learning for image recognition

Convolutional neural networks are a type of deep artificial neural network very effective for computer vision tasks, such as image classification, among other applications. These networks have a feature extraction phase followed by fully connected layers that perform the corresponding classification.

There are numerous pre-trained convolutional network architectures (Model Zoo - Deep learning code and pretrained models for transfer learning, educational purposes, and more., 2022) to recognize images, which are directly available for use in libraries such as Keras (Keras, s.f.), Theano, Pytorch (Pytorch, s.f.) They were trained with large-scale image sets, including millions of images, to allow generic models to be created. These models correctly classify the images into various categories, which represent classes of objects found in everyday life (animal species, household objects, vehicles, etc.). They also show a great ability to generalize images outside the source dataset. However, if the target problem is very different, the obtained prediction could be weak.

Different strategies are used to build a model from a pre-trained one. The most used are as follows (Figure 1 illustrates the three strategies):

- *Strategy 1*. The whole model is trained. In this case, the architecture of the pre-trained model is used and trained with the dataset. The model learns from scratch, so it will need a large data set and a lot of computational power.
- *Strategy 2*. Some layers are trained, and the others are left frozen. The initial layers of the model capture general characteristics, independent of the problem, while the final layers represent specific, more problem-dependent characteristics. This dichotomy is balanced by choosing different sectors of the network to make the tunings. Some layers are kept frozen, i.e. they do not change during training. Usually, with a small dataset and many parameters, more layers will be left frozen. Conversely, if the dataset is large and the number of parameters is small, you can improve the model by training more layers for the new task.
- *Strategy 3*. Freeze the convolutional base. This case corresponds to an extreme situation of training/freezing exchange. The main idea is to keep the convolutional base in its original form and then use its outputs to feed the classifier. The pre-trained model is used as a fixed-feature extraction mechanism, which is useful when having little computational power, the set of data is small and/or the pre-trained model solves a problem very similar to the target problem.

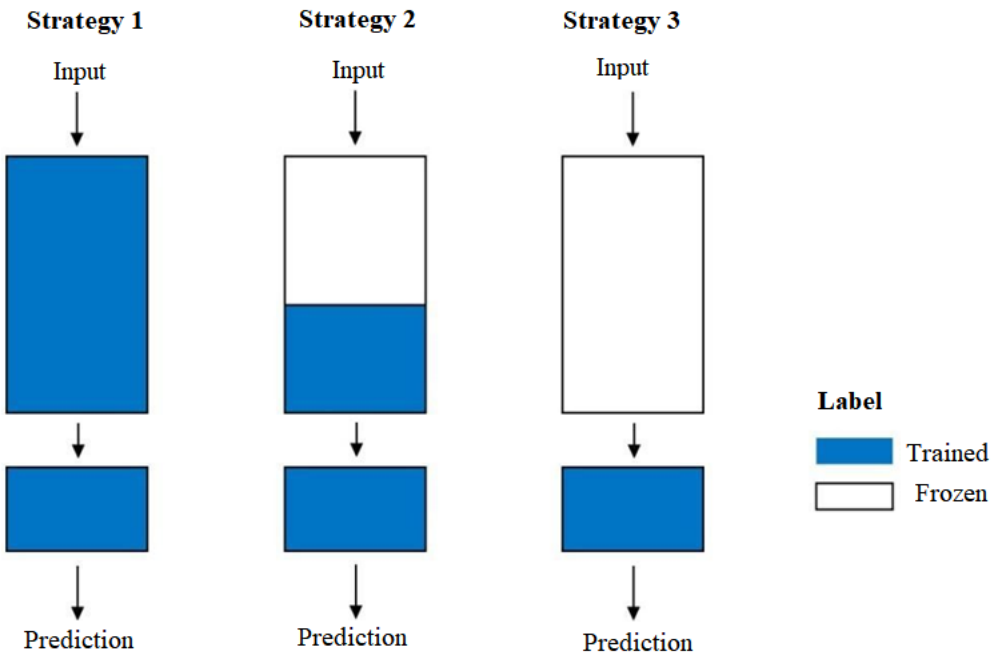

Figure 1. Tuning strategies on models

Unlike strategy 3, which is simple to implement, strategy 1 and strategy 2 require careful management of the learning rate used in the convolutional part. This hyperparameter controls the rate at which the weights of the network are updated. When using a pre-trained model based on convolutions, it is advisable to use a small learning rate, because otherwise there are high risk of losing prior knowledge. Assuming the pre-trained model performs well, maintaining a small learning rate will ensure that network weights are not abruptly distorted.

2.3 Transfer Learning Process

From a practical perspective, the entire transfer learning process can be summarized in the following stages:

- Selection of a pre-trained method. From the wide range of available models, one should be chosen that suits the problem.
- Re-training design. The problem is classified considering the size of the dataset and its similarity to the data with which the selected model was trained. This classification will allow defining the most appropriate training strategy.
- Model tuning. Re-training the model according to the strategy defined in the previous step.

Figure 2 shows a diagram with four quadrants that guide the selection of the model's training strategy. This map allows to classify the problem taking into account the size and similarity of the available data with the dataset used for the pre-trained model. A dataset

is considered small if it has less than a thousand images per class. About the similarity, common sense should prevail.

The quadrants defined on the map are as follows:

Quadrant $\boldsymbol{D^+S^-}$. Large dataset, but different from the pre-trained model dataset. This situation leads to Strategy 1. The model can be trained from scratch, taking advantage of having a large dataset. Despite the dataset difference, in practice, it can still be useful to initialize the model from a previously trained model, using its architecture. and weights.

Quadrant $\boldsymbol{D^+S^+}$. Large dataset and similar to the pre-trained model dataset. This is the ideal situation. Either of the options works. The most efficient option is Strategy 2. Since the datasets are similar, effort of learning can be avoided by leveraging prior knowledge. Therefore, it is enough to train the classifier and the final layers of the convolutional base.

Quadrant $\boldsymbol{D^-S^-}$. Small dataset and different from the pre-trained model dataset. This is the worst situation. In this case, Strategy 2 should be chosen. It is difficult to find a balance between the number of layers for training and for freezing. More training should be done, and data augmentation techniques considered.

Quadrant $\boldsymbol{D^-S^+}$. Small dataset, but similar to the pre-trained model dataset. In this case, Strategy 3 should be chosen. It is necessary to remove the last fully connected layer (classifier), run the model previously trained as a fixed feature extractor and then use the features resulting for training a new classifier.

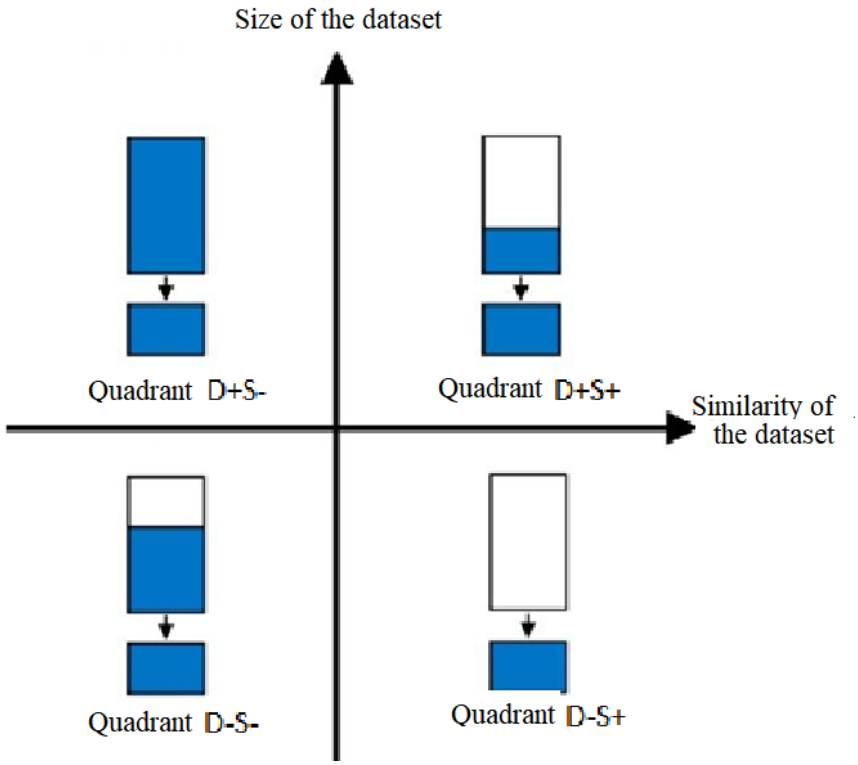

Figure 2. Decision map to fine-tune pre-trained models

3 Case Study: The Food Bank

This section describes each of the steps taken to build the image recognition model for the food bank of La Plata. The objective of this case study is to verify the feasibility of building a machine learning model which allows to properly classify food received in donations but counting with very few images to train the model.

3.1 Description of the target problem

The food bank named "Banco Alimentario de La Plata" is a non-profit civil society founded in 2000. It follows an operation process that is based on international experiences, which aims to value food. It is based on recovering food and making it available to vulnerable sectors. Food is a delicate issue, so the Food Bank has strict protocols and standards in all the processes it performs, which are reviewed through regular audits.

The Food Bank receives donations containing different products, of different brands which must be classified, stored, and recorded. Figure 3 shows one of the donations received. This task, currently performed manually, is repetitive, tedious, and error-prone, so it would be a great contribution to automate it allowing the members of the bank to focus their efforts on other tasks.

Therefore, it has been proposed to create software tools based on machine learning to assist in the task of classifying and registering products.

Figure 3 - One of the donations received by the Bank

3.2 Dataset definition

The dataset was created with images of food products grouped into different categories, such as water bottles, rice packages, tuna cans, flour packages and milk boxes. Figure 4 shows an example image for each of these categories.

A balanced dataset of 150 images was constructed, 30 images for each category, which will be separated for the training, validation, and testing sets. They were photographed with white background.

Figure 4. Dataset images for each of the categories

3.3 Image pre-processing and data augmentation

To increase the size of the dataset, the data augmentation technique was used (Shorten & Khoshgoftaar, 2019) which consists of applying geometric changes to all images in the dataset as cuts, rotations, etc. This is useful for the case of having limited data as it increases the diversity of the final set. It also helps avoiding overfitting and allows better generalization.

The data augmentation process was performed using Keras' image generator, named ImageData Generator. It was configured to normalize the values, and to generate in addition to the original image, other images transformed by zoom, horizontal and vertical flip rotations, cut transformations, etc. Figure 5 shows some of the generated images.

Figure 5: Images of milk after applying data augmentation.

This process was applied only on the training set. For each original photo, 42 new photos were obtained with the modifications mentioned above. The validation and testing package remained unchanged.

4 Transfer learning process: experiments and results

4.1 Selection of Pre-Trained Models

The first step in the transfer learning process is the selection of a pre-trained model appropriate to the problem. It is necessary to find models that offer good performance in the source tasks. In this case, several open access Deep Learning architectures published in Keras (https://keras.io/api/applications/) were investigated. This site provides a collection of Deep learning models that were previously trained using datasets available on the Internet. The selected networks were VGG16, Xception and MobileNetV2 which are very popular and were trained on the ImageNet dataset (https://www.image-net.org/) which is a database containing more than 14 million tagged images, belonging to 1000 classes, with similarity to the target problem.

Table 1 shows the characteristics of the chosen networks. The size column indicates the network disk size expressed in MB. The top1 and top5 columns refer to how often the real labels appear between the most important K predictions using the ImageNet dataset. The parameter column specifies the number of network parameters while the depth refers to the topological depth of the network, including activation layers, normalization layers, etc.

Table 1: Characteristics of the networks selected from Keras.

Model	Size (MB)	Top-1 Accuracy	Top-5 Accuracy	Parameters	Depth	Time (ms) per inference step (CPU)	Time (ms) per inference step (GPU)
VGG16	528	71.3%	90.1%	138.4M	16	69.5	4.2
Xception	88	79.0%	94.5%	22.9M	81	109.4	8.1
MobileNet	16	70.4%	89.5%	4.3M	55	22.6	3.4

4.2 Initial performance of selected pre-trained models

To describe the initial accuracy of the networks in the classification of the bank's products, the terms top-1 and top-5 are used. The term top-1 refers to the class predicted by the network. The term top-5 refers to the first 5 classes predicted by the network.

Table 2 shows the prediction provided by the network for the dataset images and the number of times it occurs. In addition, the most frequently predicted class is displayed among the classes present in the top5. For example, XCeption correctly recognizes 29 images of a water bottle (water_bottle), while only 1 is misclassified as pop_bottle.

Table 2: Predictions made by pre-trained models

	Tuna	**Flour**	**Water**	**Rice**	**Milk**
VGG 16 Top-1	oil_filter : 9 can_opener : 6 face_powder : 5 Crock_Pot : 2 Sunscreen : 2 mortar : 1 coffee_mug : 1 pill_bottle : 1 bucket : 1 tennis_ball : 1 drum : 1	Packet : 12 Sunscreen : 6 paper_towel : 4 oil_filter : 3 toilet_tissue : 2 eggnog : 1 Carton : 1 plastic_bag : 1	water_bottle : 28 Nipple : 2	Packet : 22 water_bottle : 1 Sunscreen : 1 SARONG : 1 sleeping_bag : 1 plastic_bag : 1 tennis_ball : 1 shopping_basket : 1 Band_Aid : 1	Band_Aid : 13 Packet : 4 Sunscreen : 4 NIPPLE : 3 Lotion : 3 Carton : 1 oil_filter : 1 combination_lock : 1
Most predicted class among the Top-5	oil_filter : 21	Packet : 24	water_bottle : 30	Packet : 28	Band_Aid : 28
XCeption Top1	oil_filter : 10 can_opener : 7 face_powder : 7 Crock_Pot : 3 bucket : 1 steel_drum : 1 Sunscreen : 1	Packet : 19 Sunscreen : 4 toilet_tissue : 3 Carton : 3 book_jacket : 1	water_bottle : 29 pop_bottle : 1	Packet : 27 paper_towel : 1 sleeping_bag : 1 plastic_bag : 1	Packet : 9 Band_Aid : 5 Sunscreen : 5 NIPPLE : 4 Carton : 4 Lotion : 2 web_site : 1
Most predicted class among the Top-5	oil_filter : 24	Packet : 29	water_bottle : 30	Packet : 29	Packet : 27
Mobile Top 1	oil_filter : 10 face_powder : 6 Sunscreen : 4 Bucket : 3 can_opener : 2 Crock_Pot : 1 buckle : 1 Barrel : 1 drum : 1	Packet : 20 toilet_tissue : 3 Carton : 3 oil_filter : 2 Band_Aid : 1 paper_towel : 1	water_bottle : 24 pop_bottle : 5 NIPPLE : 1	Packet : 27 sleeping_bag : 2 Jersey : 1	Packet : 14 Carton : 6 NIPPLE : 4 Band_Aid : 2 oil_filter : 1 CD_player : 1 Chihuahua : 1 Sunscreen : 1

	chocolate_willow : 1				
Most predicted class among the Top-5	face_powder : 17	Packet : 28	water_bottle : 30	Packet : 29	Packet : 27

It can be seen that some items are not recognized by the pre-trained network. For example, none of the nets properly recognize tuna cans. However, they all recognize water bottles quite accurately. In the cases of flour, rice and milk, the images are mostly recognized as the same class: package.

4.3 Re-training design

In this case study there is a small labeled dataset which is quite similar to the source problem, therefore it is necessary to explore the quadrants $\boldsymbol{D^{-}S^{+}}$ and $\boldsymbol{D^{-}S^{-}}$ described in section 3. Therefore, the pre-trained model will be used as a feature extractor of the images to be recognized and then the resulting features will be used to train a new classifier. To improve the fit, some inner layers of the pre-trained net will also be defrosted and retrained. Finally, since the data set is very small, some data augmentation technique should be applied.

4.4 Re-trainig after replacing the classification layer

This section explores the quadrant $\boldsymbol{D^{-}S^{+}}$ For that, the last layer is removed, and the previously trained model is used as a fixed characteristics extractor. Finally, the resulting features are used to train a new classifier.

The classification layer is defined as follows:

- A flatten layer
- A dense layer of X neurons
- dropout layer (0.Y)
- Dense layer with Softmax activation.
- X and Y will be different numerical values that will be analyzed for each of the models.

In all configurations, the early stopping technique is used with a maximum of 40 epochs. This technique allows training to finish once model performance stops improving over the validation dataset. This technique allows the network to have the necessary training preventing overfitting.

For each model, experiments were carried out that included analyzing the different configurations of the dropout layer, that is, different values for the variable Y, between 0.3 and 0.7. For each of these values, different configurations for the dense layer were analyzed, that is, different values for variable X. It was tested with 128, 256, 512, 1024 and 2048 neurons. Tables 3, 4 and 5 show the matrices with the training results using each combination of values, for the VGG16, Xception and MobileNetV2 networks

respectively. The results show the number of epochs the training lasted, the confusion matrix and the final accuracy of the network.

Table 3: Results of experiments conducted using the VGG16 network

Number of neurons	Dropout 0.3	Dropout 0.4	Dropout 0.5	Dropout 0.6	Dropout 0.7
128	Epoch 40/40 [9 0 0 0 0] [0 8 0 1 0] [0 0 9 0 0] [0 2 0 6 1] [0 0 0 0 9] Acc: 0.9111	Epoch 32/40 [9 0 0 0 0] [0 8 0 1 0] [0 0 9 0 0] [0 2 0 6 1] [0 0 0 0 9] Acc: 0.9111	Epoch 40/40 [9 0 0 0 0] [0 8 0 1 0] [0 0 9 0 0] [0 2 0 6 1] [0 0 0 0 9] Acc: 0.9111	Epoch 21/40 [9 0 0 0 0] [0 8 0 1 0] [0 0 9 0 0] [0 2 0 6 1] [0 0 0 0 9] Acc: 0.9111	Epoch 19/40 [9 0 0 0 0] [0 9 0 0 0] [0 0 9 0 0] [0 2 0 5 2] [0 0 0 0 9] Acc: 0.9111
256	Epoch 39/40 [9 0 0 0 0] [0 9 0 0 0] [0 0 9 0 0] [0 2 0 6 1] [0 0 0 0 9] Acc: 0.9333	Epoch 37/40 [9 0 0 0 0] [0 8 0 1 0] [0 0 9 0 0] [0 2 0 6 1] [0 0 0 0 9] Acc: 0.9111	Epoch 27/40 [9 0 0 0 0] [0 8 0 1 0] [0 0 9 0 0] [0 2 0 6 1] [0 0 0 0 9] Acc: 0.9111	Epoch 24/40 [9 0 0 0 0] [0 8 0 1 0] [0 0 9 0 0] [0 2 0 6 1] [0 0 0 0 9] Acc: 0.9111	Epoch 29/40 [9 0 0 0 0] [0 9 0 0 0] [0 0 9 0 0] [0 2 0 6 1] [0 1 0 0 8] Acc: 0.9111
512	Epoch 27/40 [9 0 0 0 0] [0 8 0 1 0] [0 0 9 0 0] [0 2 0 6 1] [0 0 0 0 9] Acc: 0.9111	Epoch 35/40 [9 0 0 0 0] [0 8 0 1 0] [0 0 9 0 0] [0 2 0 6 1] [0 0 0 0 9] Acc: 0.9111	Epoch 31/40 [9 0 0 0 0] [0 8 0 1 0] [0 0 9 0 0] [0 2 0 7 0] [0 1 0 0 8] Acc: 0.9111	Epoch 40/40 [9 0 0 0 0] [0 8 0 1 0] [0 0 9 0 0] [0 2 0 6 1] [0 0 0 0 9] Acc: 0.9111	Epoch 26/40 [9 0 0 0 0] [0 9 0 0 0] [0 0 9 0 0] [0 2 0 6 1] [0 0 0 0 9] Acc: 0.9333
1024	Epoch 25/40 [9 0 0 0 0] [0 8 0 1 0] [0 0 9 0 0] [0 2 0 7 0] [0 0 0 0 9] Acc: 0.9333	Epoch 23/40 [9 0 0 0 0] [0 8 0 1 0] [0 0 9 0 0] [0 1 0 8 0] [0 0 0 0 9] Acc: 0.9555	Epoch 26/40 [9 0 0 0 0] [0 8 0 1 0] [0 0 9 0 0] [0 2 0 6 1] [0 0 0 0 9] Acc: 0.9111	Epoch 33/40 [9 0 0 0 0] [0 8 0 1 0] [0 0 9 0 0] [0 2 0 6 1] [0 0 0 0 9] Acc: 0.9111	Epoch 32/40 [9 0 0 0 0] [0 8 0 1 0] [0 0 9 0 0] [0 2 0 6 1] [0 0 0 0 9] Acc: 0.9111
2048	Epoch 24/40 [9 0 0 0 0] [0 8 0 1 0] [0 0 9 0 0] [0 2 0 6 1] [0 0 0 0 9] acc:0.9111	Epoch 23/40 [9 0 0 0 0] [0 8 0 1 0] [0 0 9 0 0] [0 2 0 7 0] [0 1 0 0 8] Acc: 0.9111	Epoch 25/40 [9 0 0 0 0] [0 8 0 1 0] [0 0 9 0 0] [0 2 0 6 1] [0 0 0 0 9] Acc: 0.9111	Epoch 29/40 [9 0 0 0 0] [0 9 0 0 0] [0 0 9 0 0] [0 2 0 6 1] [0 1 0 0 8] Acc: 0.9111	Epoch 23/40 [9 0 0 0 0] [0 7 0 2 0] [0 0 9 0 0] [0 2 0 7 0] [0 0 0 0 9] Acc: 0.9111

Table 4: Results of experiments conducted using the Xception network

	Dropout 0.3	**Dropout 0.4**	**Dropout 0.5**	**Dropout 0.6**	**Dropout 0.7**
128	Epoch 28/40 [9 0 0 0 0] [0 9 0 0 0] [0 0 9 0 0] [0 3 0 5 1] [0 0 0 0 9] Acc: 0.9111	Epoch 18/40 [9 0 0 0 0] [0 9 0 0 0] [0 0 9 0 0] [0 2 0 6 1] [0 0 0 0 9] Acc:0.9333	Epoch 27/40 [9 0 0 0 0] [0 9 0 0 0] [0 0 9 0 0] [0 2 0 7 0] [0 0 0 0 9] Acc: 0.9556	Epoch 28/40 [9 0 0 0 0] [0 9 0 0 0] [0 0 9 0 0] [0 2 1 5 1] [0 0 0 0 9] Acc: 0.9111	Epoch 14/40 [9 0 0 0 0] [0 8 0 1 0] [0 0 9 0 0] [0 2 0 6 1] [0 0 0 0 9] Acc: 0.9111
256	Epoch 17/40 [9 0 0 0 0] [0 9 0 0 0] [0 0 9 0 0] [0 3 0 5 1] [0 0 0 0 9] Acc: 0.9111	Epoch 21/40 [9 0 0 0 0] [0 9 0 0 0] [0 0 9 0 0] [0 3 0 5 1] [0 0 0 0 9] Acc: 0.9111	Epoch 21/40 [9 0 0 0 0] [0 9 0 0 0] [0 0 9 0 0] [0 2 0 6 1] [0 0 0 0 9] Acc: 0.9333	Epoch 16/40 [9 0 0 0 0] [0 9 0 0 0] [0 0 9 0 0] [0 2 0 7 0] [0 0 0 0 9] Acc: 0.9556	Epoch 40/40 [9 0 0 0 0] [0 9 0 0 0] [0 0 9 0 0] [0 3 0 6 0] [0 0 0 0 9] Acc: 0.9333
512	Epoch 29/40 [9 0 0 0 0] [0 9 0 0 0] [0 0 9 0 0] [0 3 0 6 0] [0 0 0 0 9] Acc: 0.9333	Epoch 20/40 [9 0 0 0 0] [0 8 0 1 0] [0 0 9 0 0] [0 2 0 7 0] [0 0 0 0 9] Acc: 0.9333	Epoch 15/40 [9 0 0 0 0] [0 9 0 0 0] [0 0 9 0 0] [0 3 0 6 0] [0 1 0 0 8] Acc: 0.9111	Epoch 20/40 [9 0 0 0 0] [0 9 0 0 0] [0 0 9 0 0] [0 2 0 7 0] [0 0 0 0 9] Acc: 0.9556	Epoch 22/40 [9 0 0 0 0] [0 9 0 0 0] [0 0 9 0 0] [0 3 0 5 1] [0 0 0 0 9] Acc: 0.9111
1024	Epoch 17/40 [9 0 0 0 0] [0 9 0 0 0] [0 0 9 0 0] [0 2 0 7 0] [0 0 0 0 9] Acc: 0.9555	Epoch 18/40 [9 0 0 0 0] [0 9 0 0 0] [0 0 9 0 0] [0 2 0 7 0] [0 0 1 0 8] Acc: 0.9333	Epoch 27/40 [9 0 0 0 0] [0 8 0 1 0] [0 0 9 0 0] [0 3 0 6 0] [0 0 0 0 9] Acc: 0.9111	Epoch 28/40 [9 0 0 0 0] [0 9 0 0 0] [0 0 9 0 0] [0 3 0 5 1] [0 0 0 0 9] Acc: 0.9111	Epoch 11/40 [9 0 0 0 0] [0 9 0 0 0] [0 0 9 0 0] [0 3 0 5 1] [0 0 0 0 9] Acc: 0.9111
2048	Epoch 14/40 [9 0 0 0 0] [0 9 0 0 0] [0 0 9 0 0] [0 3 0 5 1] [0 0 0 0 9] Acc: 0.9111	Epoch 28/40 [9 0 0 0 0] [0 9 0 0 0] [0 0 9 0 0] [0 2 0 7 0] [0 0 0 0 9] Acc: 0.9555	Epoch 26/40 [9 0 0 0 0] [0 9 0 0 0] [0 0 9 0 0] [0 3 0 5 1] [0 0 0 0 9] Acc: 0.9111	Epoch 16/40 [9 0 0 0 0] [0 9 0 0 0] [0 0 9 0 0] [0 3 0 6 0] [0 0 0 0 9] Acc: 0.9333	Epoch 28/40 [9 0 0 0 0] [0 8 0 1 0] [0 0 9 0 0] [0 2 0 7 0] [0 0 0 0 9] Acc: 0.9333

Table 5: Results of experiments conducted using the MobileNetV2 network

	Dropout 0.3	**Dropout 0.4**	**Dropout 0.5**	**Dropout 0.6**	**Dropout 0.7**
128	Epoch 24/40 [9 0 0 0 0] [0 8 0 1 0] [0 0 9 0 0] [0 2 0 7 0]	Epoch 18/40 [9 0 0 0 0] [0 8 0 1 0] [0 0 9 0 0] [0 0 0 8 1]	Epoch 21/40 [9 0 0 0 0] [0 9 0 0 0] [0 0 9 0 0] [0 2 0 6 1]	Epoch 20/40 [9 0 0 0 0] [0 9 0 0 0] [0 0 9 0 0] [0 1 0 8 0]	Epoch 28/40 [9 0 0 0 0] [0 8 0 1 0] [0 0 9 0 0] [0 1 0 8 0]

	[0 0 0 0 9] Acc: 0.9333	[0 0 0 0 9] Acc: 0.9555	[0 0 0 0 9] Acc: 0.9333	[0 0 0 0 9] Acc: 0.9777	[0 0 0 0 9] Acc: 0.9555
256	Epoch 21/40 [9 0 0 0 0] [0 8 0 1 0] [0 0 9 0 0] [0 1 0 8 0] [0 0 0 2 7] Acc: 0.9111	Epoch 20/40 [9 0 0 0 0] [0 9 0 0 0] [0 0 9 0 0] [0 2 0 6 1] [0 0 0 0 9] Acc: 0.9333	Epoch 20/40 [9 0 0 0 0] [0 9 0 0 0] [0 0 9 0 0] [0 1 0 8 0] [0 0 0 0 9] Acc: 0.9777	Epoch 13/40 [9 0 0 0 0] [0 8 0 1 0] [0 0 9 0 0] [0 1 0 6 2] [0 0 0 0 9] Acc: 0.9111	Epoch 16/40 [9 0 0 0 0] [0 9 0 0 0] [0 0 9 0 0] [0 2 0 6 1] [0 0 0 0 9] Acc: 0.9333
512	Epoch 15/40 [9 0 0 0 0] [0 8 0 1 0] [0 0 9 0 0] [0 2 0 7 0] [0 0 0 0 9] Acc: 0.9333	Epoch 12/40 [9 0 0 0 0] [0 8 0 1 0] [0 0 9 0 0] [0 0 0 9 0] [0 0 0 0 9] Acc: 0.9777	Epoch 23/40 [9 0 0 0 0] [0 8 0 1 0] [0 0 9 0 0] [0 0 0 9 0] [0 0 0 0 9] Acc: 0.9777	Epoch 31/40 [9 0 0 0 0] [0 9 0 0 0] [0 0 9 0 0] [0 2 0 7 0] [0 0 0 0 9] Acc: 0.9555	Epoch 30/40 [9 0 0 0 0] [0 9 0 0 0] [0 0 9 0 0] [0 2 0 7 0] [0 0 0 0 9] Acc: 0.9555
1024	Epoch 27/40 [9 0 0 0 0] [0 9 0 0 0] [0 0 9 0 0] [0 0 0 8 1] [0 0 0 0 9] Acc: 0.9777	Epoch 20/40 [9 0 0 0 0] [0 9 0 0 0] [0 0 9 0 0] [0 2 0 7 0] [0 0 0 0 9] Acc: 0.9555	Epoch 17/40 [9 0 0 0 0] [0 9 0 0 0] [0 0 9 0 0] [0 2 0 5 2] [0 0 0 0 9] Acc: 0.9111	Epoch 16/40 [9 0 0 0 0] [0 8 0 1 0] [0 0 9 0 0] [0 0 0 9 0] [0 0 0 0 9] Acc: 0.9777	Epoch 21/40 [9 0 0 0 0] [0 8 0 1 0] [0 0 9 0 0] [0 0 0 9 0] [0 0 0 0 9] Acc: 0.9777
2048	Epoch 25/40 [9 0 0 0 0] [0 8 0 1 0] [0 0 9 0 0] [0 1 0 8 0] [0 0 0 1 8] Acc: 0.9333	Epoch 30/40 [9 0 0 0 0] [0 9 0 0 0] [0 0 9 0 0] [0 2 0 7 0] [0 0 0 0 9] Acc: 0.9555	Epoch 30/40 [9 0 0 0 0] [0 9 0 0 0] [0 0 9 0 0] [0 1 0 8 0] [0 0 0 0 9] Acc: 0.9777	Epoch 24/40 [9 0 0 0 0] [0 8 0 1 0] [0 0 9 0 0] [0 2 0 7 0] [0 0 0 0 9] Acc: 0.9333	Epoch 13/40 [9 0 0 0 0] [0 9 0 0 0] [0 0 9 0 0] [0 0 0 8 1] [0 0 0 0 9] Acc: 0.9777

It is noteworthy that all models show good accuracy recognizing water bottles and cans of tuna. This is probably because the shape of these products is evidently different from the others. On the other hand, in the cases of flour, models often are not able to recognize the totality of the images.

Regarding the performance of the networks, they all obtain very good results. There is no significant difference between the different configurations. The best model is obtained with MobileNet, which reaches an accuracy of 0.97% with only 13 training epochs.

4.5 Re-training by defrosting some layers

To explore the quadrant $\boldsymbol{D^{-}S^{-}}$, some layers of the models were defrosted and re-trained.

In the case of VGG16, layers are defrosted from the 4th convolution block. This means moving from having 12,848,133 trainable parameters in the dense layer, to 25,827,333 trainable parameters and 1,735,488 non-trainable.

In the case of Xception there are 51,383,301 trainable parameters in the dense layer and when defrosting the layers from block 7 you have 67,855,541 trainable parameters and 4,389,240 non-trainable

Finally, in the case of MobileNet V2 there are 32,378,373 trainable parameters in the dense layer and when defrosting the layers from block 9 you have 34,184,197 trainable parameters and 189,504 non-trainable.

After applying this process, the results obtained were similar. The tuned models were not able to improve the 97% accuracies already obtained.

5 Conclusion and future work

In this article, the process of building a model for image recognition and classification using the transfer learning technique, was detailed. The research was based on addressing a particular case study, the income of donations to the Food Bank of La Plata. One of the determining features of the domain was that the available training dataset was very small.

Each stage of model development was analysed. The first stage consisted of investigating the pre-trained models available for similar tasks and thus selecting the candidates that best suit the target problem. Three pre-trained models of different characteristics were tested: VGG16, Xception and MobileNetV2.

The re-training process of these candidate models was then designed, choosing the most appropriate training strategies taking into account the size of the dataset and the similarity between source and target domains.

Finally, the candidate models were re-trained by applying the strategies defined in the previous step and evaluating different configurations of dropout layers and number of neurons. Controlled tests were performed under the same conditions.

From this analysis it was possible to determine the pre-trained model that acquires knowledge faster and achieves better performance.

The best accuracy was obtained with MobileNet, which reaches 0.97% with only 13 training times. However, no significant difference was observed between the different configurations since all models achieved very good results, greater than 0.90% and in all cases without exceeding 40 training epochs.

The results obtained in the investigations corroborate that the techniques of transfer learning in deep neural networks are appropriate to solve tasks of detection and classification of images even in cases where there is a very moderate number of samples.

Future developments: This work can be extended by incorporating other models to the analysis and also evaluating different techniques of re-training and adjustment, additional to those developed so far. On the other hand, the final model developed for the food bank should be integrated into the software applications currently used in the bank for entry and recording of donations. This task was started but there are still several tasks to be done until the system can be put into operation.

References

Alpaydin, E. (2016). *Machine Learning: The New AI.* MIT Press.

Anaconda. (s.f.). Obtenido de https://www.anaconda.com/

Goodfellow , I., Bengio, Y., & Courville, A. (2016). *Deep Learning.* Adaptive Computation and Machine Learning series.

Han, X. Z. (2021). Pre-Trained Models: Past, Present and Future,. *AI Open, https://doi.org/10.1016/*.

Hugging-Face. (s.f.). *Transformers*. Recuperado el 2022, de https://github.com/huggingface/transformers

Jialin , S., & Yang, Q. (2010). A Survey on Transfer Learning. *IEEE Transactions on Knowledge and Data Engineering, (22)10.* , 1345-1359.

Keras. (s.f.). Obtenido de Keras https://keras.io/

Krizhevsky, A., Sutskever, N., & Hinton, G. (2017). ImageNet classification with deep convolutional neural networks. *Communications of the ACM, 60(6), . doi:10.1145/3065386*, 84–90.

LeCun, Y., Bengio, Y., & Hinton, G. (2015). Deep learning. *Nature 521*, 436-444.

Model Zoo - Deep learning code and pretrained models for transfer learning, educational purposes, and more. (2022). Obtenido de https://modelzoo.co/(date

Neyshabur, B. S. (2020). What is being transferred in transfer learning?. . *Advances in neural information processing systems, 33*, 512-523.

Pytorch. (s.f.). Obtenido de https://pytorch.org/

Russell, S., & Norvig, P. (2021). *Artificial Intelligence. A Modern Approach. Copyright © 2021* . Pearson Education. 4ta edición.

Shorten, C., & Khoshgoftaar, T. M. (2019). A survey on Image Data Augmentation for Deep Learning. *Mathematics and Computers in Simulation.r. 6: 60*. doi:10.1186/s40537-019-0197-0

TensorFlow. (s.f.). Obtenido de https://www.tensorflow.org/

Torrey, L., & Shavlik, J. (2009). *Transfer Learning. In Machine Learning Applications and Trends: Algorithms, Methods, and Techniques. p.834. University of Wisconsin.* University of Wisconsin.