Open Peer Review on Qeios

Development of a Mobile Application for Flag Identification based on Artificial Neural Networks

Ivan Zatezalo[1], Ivan Dunđer[1]

1 University of Zagreb

Funding: The author(s) received no specific funding for this work.

Potential competing interests: The author(s) declared that no potential competing interests exist.

Abstract

Machine learning, and more specifically its subfield known as deep learning, have been driving new disruptive technologies and recent accomplishments in various industries. This paper applies to some extent the same technology to develop a mobile application for educational purposes, that allows one to identify unknown flags with the help of artificial intelligence. For the application to be functional and effective, a custom dataset had to be created and processed in various ways in order to provide a collection of data that resembles data from the real world. The collection would then be used as the input data for constructing the model, which is developed within the framework TensorFlow. Once the model was developed, it was implemented as part of a mobile application programmed with Flutter. The functionality of the mobile application and the model's accuracy are then put to the test against over a hundred new images the model has not seen previously or been trained on. The result of this evaluation could be considered an estimate of the readiness and usability of the application in a real-world scenario.

Keywords: convolutional neural networks, mobile application, TensorFlow, object identification, machine learning, deep learning, artificial intelligence.

1. Introduction and Motivation

The use of artificial intelligence is truly wide, but one of its main goals is to make human life easier and better. In order for machines to be able to understand and grasp the "meaning" of data, it is necessary to use various machine learning algorithms that make this process feasible. Machine learning allows us to understand the data in a way where its future predictions will be based on the previous data. This way if a behavior occurs again, machines will, through programmed solutions for each occurrence, be able to react and give the appropriate solutions for the task at hand. With this goal in mind, the development of a mobile application called "FlagAFlag" began.

The main goal of this research was to make an educational mobile application that would help users identify, i.e., classify flags they found around the world with the help of a camera. To create such an application, it was necessary to develop artificial intelligence that would be able to accomplish such a task. This process required collecting the data and then augmenting it in a way so that it was as similar as possible to data in the real world. The process of identification of particular patterns is possible due to the model – a piece of programming code that with help of machine learning is able to make assumptions based on the data it has seen before (Cowley, Radich, 2019). This process allows the application “FlagAFlag” to make future predictions and help users correctly classify their requests or, in this case, particular flags they cannot identify, regardless if the flags derive from ready-made and stored images or from taking photographs with a camera.

With these necessary steps in mind, the following research questions were raised:

1. Which steps were necessary when collecting and augmenting the dataset in order for it to be reflective of the data present in the real world?
2. How challenging was the process of tuning the artificial neural network in order for it to understand the given data?
3. For what reason did the mobile application misclassify the selected image or captured photograph during the development, and why did the following occur?
4. How exactly could potential classification problems that may arise be resolved?

The main hypothesis in this paper is that the position of the flag affects the accuracy of the machine-trained model, and hence, the efficacy of the developed application.

2. Literature Review

When it comes to machine learning in general, today there are different types of machine learning algorithms that are organized into a taxonomy that is based on the desired outcome of the algorithm. Some of the common algorithm types include supervised learning, unsupervised learning, semi-supervised learning, reinforcement learning, transduction, and learning to learn algorithm which learns its own inductive bias based on previous experience (Zhang, 2010).

On the other hand, deep learning is a particular kind of machine learning that achieves immense power and flexibility by learning to represent the world as a nested hierarchy of concepts and representations, whereas each concept is defined in relation to simpler concepts, and more abstract representations are computed in terms of less abstract ones (Goodfellow et al., 2016).

Machine learning today is intensively applied in a wide range of sciences and for various purposes and different tasks. For instance, machine learning can be used for the automatic recognition of information behavior (Lugović, Dunđer, 2017),

which can affect how information systems should be designed. It can also be applied in the process of detecting affective states in speech (Lugović et al., 2016) or text (Dunđer, Pavlovski, 2019a), which can summarize expressed emotions in a given setting. It has also a significant application in natural language processing, e.g., in vector analyses of digital corpora (Dunđer, Pavlovski, 2019b) where it can detect semantic similarities between entities within a dataset.
In the context of the Croatian language and available tools and resources, machine learning has been especially explored with a special focus on machine translation (Dunđer, 2020; Dunđer et al., 2020) and document classification (Dunđer et al., 2015).

More specifically on the topic of deep learning, one of the most prominent algorithms for optimization in deep learning is *Adam*. This algorithm was introduced by Kingma and Ba (2015). This paper presented its implementation which is straightforward. Adam was shown to be computationally efficient, required a small amount of memory, and was shown to be well suited for problems that are large in terms of data or parameters. Empirical results in this paper demonstrated that Adam worked well in practice and compared favorably to other stochastic optimization methods.

A research paper proposed an image dataset that was organized according to the WordNet hierarchy (Deng et. al., 2009). The project was inspired by two important needs in computer vision research. The first issue was a growing demand for a high-quality object categorization benchmark with clearly established evaluation metrics. The second issue computer vision faced at the time was the need for more data to enable more generalizable machine learning methods. For these problems to be resolved it was crucial to provide researchers with a large-scale image database for both training and testing.

On the subject of object identification and classification, research into the topic of national flag classification has been done by Lodh and Parekh (2016). This research fed images into a k-NN classifier for its class discrimination. Its dataset contained 400 images with 200 classes, i.e., national flags present. This type of classification yielded an accuracy of 99% (Lodh, Parekh, 2016).

One research proposed a system that would locate and segment a flag from the surrounding area in a photo. Its dataset contained 186 different classes with 36 samples per class. For classification to work its users are asked to select a region of interest by cropping the flag. Its identification is not automatic but rather it lists the countries based on color similarity (Hart et al., 2004).

Another recent research contained over 20,000 images with 100 images per class, i.e., country. Its Convolutional Neural Network (CNN) reached an average precision of 89.5%. This research noted challenging conditions, such as deformation and occlusion (Said, Barr, 2021).

In research conducted by Mahamd et al. (2021), transfer learning technology was used to identify the nationality of eight countries based on a custom dataset. The custom dataset included countries from The South Asian Association for

Regional Cooperation (SAARC).

The goal of this research was to create a classifier as a way for people of non-SAARC countries to identify the national flag in question. The dataset contained 4,000 images. This model yielded an accuracy of 96.6%.

Deep learning methods were also explored as a way to automatically classify and detect plant diseases from leaf images. This research described the complete process, respectively, from collecting the images used for training and validation to image preprocessing and augmentation and finally the procedure of training the deep CNN and fine-tuning. Its database was extended to more than 30,000 images using appropriate transformations. The experimental results achieved accuracy between 91% and 98%, for separate class tests. The final overall accuracy of the trained model was 96.3% (Sladojevic et al., 2016).

The application of neural networks is also present in the classification of music genres. The research implementation involved taking songs, converting them into short-time segments, and representing the time segments by their respective spectrogram images. Each of these spectrograms was labeled by music genre and then the respective inputs were fed into a CNN. The neural network had six convolutional layers followed by a fully connected layer and then a *Softmax* function at the end in order to calculate the probability of each genre detected and a one-hot array of genre classifications. The results showed an accuracy of 85% on the given test data (Pelchat, Gelowitz, 2020).

3. Research and Development of the Mobile Application

The following subsections describe the process of dataset acquisition, selection, and adaptation, the construction of the model with the help of machine learning and a specific framework, and the development of the mobile application itself.

What the research in this paper aims to do is to successfully implement a model within a mobile application, while still achieving high accuracy. As well as the problem of implementation, this paper goes to show potential hurdles that occurred while collecting and carefully selecting the gathered dataset to achieve the highest possible accuracy. Just as the research conducted by Said and Barr (2021) noted challenging conditions such as deformation, the research done in this paper aims to further expand this by showing how crucial the shade of color is to the correct classification of the national flag.

3.1 Dataset acquisition, selection, and adaption

One of the many challenges in machine learning is data collection. The challenge itself stems from the need for data to be properly prepared, analyzed, processed, and adapted for a selected domain and task. This can often be a problem due to the fact that machine learning is based on, both, data quality and quantity, and if the researched domain or task is relatively new there will simply be a lack of required data (Roy et al., 2019). Since the task of flag identification, i.e., classification is quite a niche, there are no publicly available and organized datasets that contain realistic images. For that reason, a completely new and custom dataset had to be developed from scratch.

During the data collection process, it was essential to understand and develop the collection from the user's perspective, and to take into account various interactions and obstacles that the user might encounter in a real-world scenario. Given that the goal of this research was to develop a mobile application that would help users classify flags, it was important to recognize what kind of requests and inputs would its users provide to the application.

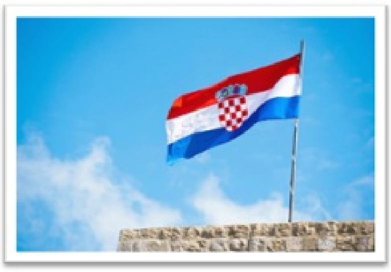


Figure 1. *A representation of a suitable image for the development of the custom dataset.*

Source: *William-Ellis (n.d.)*

Images similar to the one present in Figure 1 were used to create the custom dataset as they were very akin to images that users would end up feeding into the application. A situation in which a user would give the application an image like the one present in Figure 2 seemed far more unlikely compared to the one shown in Figure 1.

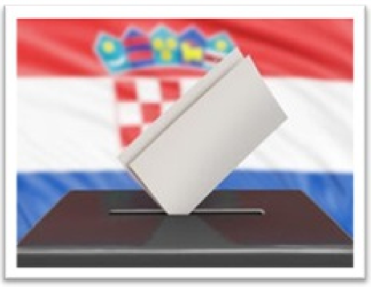


Figure 2. *A representation of a substandard image for the development of the custom dataset.*

Source: *Depositphotos (n.d.)*

Continuing from the user's perspective, it was important to recognize that images would not be perfect and could contain different irrelevant objects, rotations, different light levels, and characteristics alike. A well-trained model is one that contains data with such characteristics because it has taken into account the unpredictability of future requests propped by users. This is very significant as the goal of machine learning is to develop a robust model that is able to process the user's requests with high accuracy.

The iteration present in this paper contains 48 national flags, i.e., 48 different classes of flags. Because the selection process was so time-consuming it was not possible at the moment of writing this paper to include the flags of all nationalities in the world. For this reason, the number 48 was chosen as it is representative of countries present on the European continent.

The process of gathering the images was automated via a Python program. The Python script greatly reduced the amount of time needed to acquire the data. Besides the script being written in Python, it contained packages such as "BeautifulSoup", "Selenium" and "urllib" in order to enable the unsupervised automation of the downloading process. Once the program has finished, the result was a collection of over 16,000 images or ca. 350 images per class.

3.1.1. Data selection and processing

Given that approximately 350 images were acquired during the process for each country or class, it was very likely that some of the images would not be suitable for the model development due to poor quality or the details present inside an image. Those images had to be filtered out manually. As mentioned before, it was essential to create a realistic dataset so that the model itself had the highest possible accuracy when applied in the real world. The result of the selection of over 16,000 images was surprising. The average number of images per class dropped from ca. 350 to only ca. 40. The majority of images did not fit the criteria stated beforehand or did not contain the named flag in the image.

It is well established that in machine learning, models with more data show significantly better results, however, it is equally as important that the data is of high quality and up to a given task. For that reason, the best models are those that have a good ratio of quality and quantity of data. That is to say, if machine learning is performed on data that is of low quality the model will learn the wrong properties and, thus, its efficiency, later on, will be very low (Appen, 2018).

Given that the dataset contained 48 different classes, 40 images per flag was a very small number for an efficient model. The optimal amount of data, when developing and training a custom model, is not defined and experimentation is often needed to determine this exact number in order for the model to become robust, however, there is an unspoken rule of 1,000 images per class in the case of developing models for image classification. The number 1,000 stems from the original image classification challenge. Namely, this figure was satisfactory in the creation of early classifiers such as AlexNet, and based on this, the number 1,000 is still recommended (Warden, 2017). For this reason, the dataset was further expanded.

Upon further research, it was concluded that flags could often be found in non-static positions. To elaborate, flags are often placed in a vertical position or mirrored due to wind or other factors. So, two additional Python scripts were programmed. One script rotated the original image by 90 degrees, while the other one mirrored the original image.

Due to the generation of additional copies, the average number of images per class increased to 120. This number is not close to the recommended of 1,000, however, adding additional images in different positions will definitely improve the model itself and adapt it to real-world scenarios. Also, for the purpose of research on this topic, this was quite sufficient. After this process, the total amount of images was 5,760.

3.2. The problem of data similarity

Looking at flags from all around the world, it is obvious that various symbols and color combinations are present, however, it is just as noticeable that there are some similarities between them. These similarities are more common in countries located in the same region. Some similarities in design often represent the same cultural, political, or religious heritage. This resemblance can be seen in countries such as Croatia, Slovenia, Serbia, the Czech Republic, and Slovakia (Lee, 2017). This similarity can pose a significant problem in the case of a very small data collection and therefore the model may wrongly classify a flag.

The biggest hurdle while training and testing the model was posed by countries such as France, Luxembourg, and the Netherlands. By rotating the flag of the Netherlands, it becomes increasingly difficult to notice the difference between it and France. Another problem could be posed by the shade of blue color present inside a flag. A detailed analysis of this problem is given in later subsections.

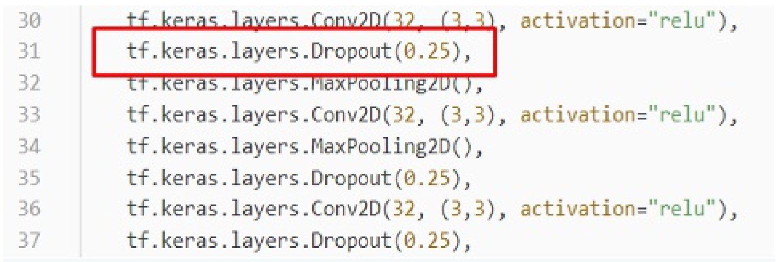
3.3. Resources

The experiments were conducted using a regular desktop computer with a Ryzen 5 3600 CPU, 16 GB of RAM, RX570 GPU, and running on a Windows 10 operating system. The artificial neural network (ANN) was developed using the *TensorFlow* framework.

Due to the incompatibility with AMD graphics cards, TensorFlow version 2.0 had to be downgraded to version 1.12 to enable the experiments on the GPU. The process of tuning and building the model took close to 45 hours in total.

3.4. Model development

With the intention of classifying images, a Convolutional Neural Network was chosen as a way to train the model. Its architecture held typical layers such as convolutional layers and pooling layers, but also dropout layers were added to prevent overfitting. The following can be seen in Figure 3.



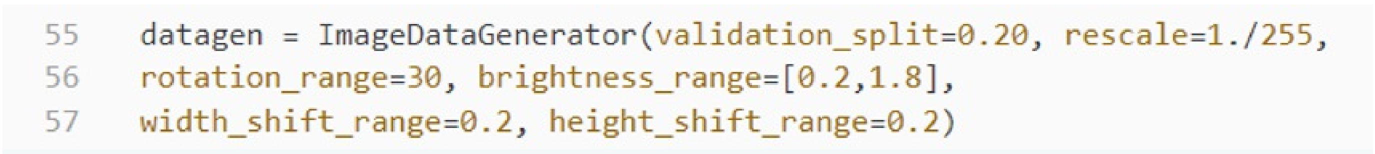
```
    tf.keras.layers.Conv2D(32, (3,3), activation="relu"),
    tf.keras.layers.Dropout(0.25),
    tf.keras.layers.MaxPooling2D(),
    tf.keras.layers.Conv2D(32, (3,3), activation="relu"),
    tf.keras.layers.MaxPooling2D(),
    tf.keras.layers.Dropout(0.25),
    tf.keras.layers.Conv2D(32, (3,3), activation="relu"),
    tf.keras.layers.Dropout(0.25),
```

Figure 3. *Application of dropout layers*

Source: *Authors*

To further prevent overfitting, the *ImageDataGenerato*r class was added with the help of the*Keras API.* Its use allowed the expansion of the dataset by supplying the model with new variations of data with every epoch.

The dataset which is described in the previous subsection was split into a ratio of 80:20. The model was trained based on 80% of the data while the remaining 20% of it was used as the validation set for subsequent performance analyses. The goal of using a validation set was to simulate real-world data in order to see the model’s accuracy on data that it has not seen before. Figure 4 displays the usage of the ImageDataGenerator class, as well as the values present within the class.



```
datagen = ImageDataGenerator(validation_split=0.20, rescale=1./255,
rotation_range=30, brightness_range=[0.2,1.8],
width_shift_range=0.2, height_shift_range=0.2)
```

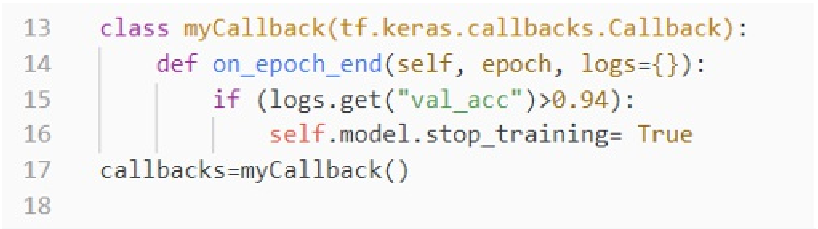
Figure 4. *Usage of ImageDataGenerator class*

Source: *Authors*

Since there were 48 different classes of flags, it was necessary to select an algorithm that would be able to display a probability value for the selected and earlier-stored images or photographs captured by a camera. This was possible with the combination of the Softmax activation function and the *Categorical Cross-Entropy* loss function. This combination, also known as *Softmax loss*, works because it trains a Convolutional Neural Network to output a probability over the classes found in the dataset for the selected and earlier-stored images or captured photographs. This combination is usually used in multi-class classification (Gómez, 2018), and therefore also employed while creating this particular model. Alongside Softmax loss, to train the model the optimizer *Adam* was chosen with a learning rate of 0.001. Some experimentation was done using other optimizers such as *Nadam*, *RMSprop,* and *SGD*, however, the best results were achieved using the Adam optimizer. Inside the hidden layer of the Convolutional Neural Network, the activation function *ReLU* (Rectified

Linear Unit) was utilized. This activation function was chosen due to its ability to overcome the vanishing gradient problem as it allowed the model to learn faster and better.

To stop the training once a certain accuracy level was reached, a class*callback* was created. The programming code for the class callback is shown in Figure 5. This class allowed the model to stop its training once the threshold of 94% accuracy of the validation set was reached. This result was achieved just around the 100-epoch mark. Having reached this goal, the training stopped, and the development of the mobile application "FlagAFlag" began.



```
class myCallback(tf.keras.callbacks.Callback):
    def on_epoch_end(self, epoch, logs={}):
        if (logs.get("val_acc")>0.94):
            self.model.stop_training= True
callbacks=myCallback()

```

Figure 5. *The programming code for stopping the training process*

Source: *Authors*

3.5. Development of the mobile application "FlagAFlag"

The next step in this research was to create a mobile application so that the trained model could be implemented. This application was developed inside the Flutter framework which uses the programming language Dart. The problem with developing mobile applications lies in the fact that when creating one application, it is necessary to create various versions for different platforms, e.g. one version of the application for iOS and another one for Android. This endeavor requires knowledge of several programming languages, as well as operating systems. The advantage of Flutter is that it provides a multi-platform solution, thus significantly shortening the development time and reducing the efforts required for successful cross-platform development of the application. Additionally, time to develop is reduced with the help of packages and plugins which work in a similar fashion to packages inside Python. One disadvantage to using a cross-platform solution such as Flutter is performance. Applications made using native development are faster and if performance is key in an application, native development is often a better choice.

3.5.1. Creating the user interface

During the process of creating the user interface (UI), the main goal was to create a simple and user-friendly interface so that its users were not overwhelmed with unnecessary information or add-ons. Since the goal of the application was to enable flag identification, it was important to allow users to take a picture of a flag within the application. Of course, it was just as important to allow users to have the option to select an image of a flag from their own gallery in case they already had an image ready on their phone.

The main screen was developed with the help of “camera” and “image_picker” plugins. To preserve simplicity, the main screen was designed to be as similar as possible to a typical camera interface. This removed the need for users to learn a completely new interface and allowed for seamless classification (see Figure 6).

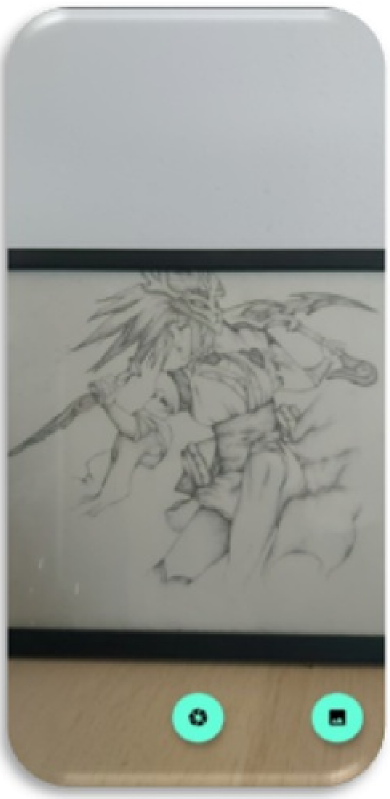


Figure 6. *The user interface within the application “FlagAFlag”*

Source: *Authors*

3.5.2. Implementation of the model

In order to successfully implement the model within Flutter, it was necessary to convert the trained model into a *TensorFlow Lite* version of the model. TensorFlow Lite enables machine learning on mobile devices, thus preventing the need to share data between users and servers on which machine learning would otherwise take place. Additionally, this removes the need for the internet as a prerequisite for image classification.

Converting models to the TensorFlow Lite version significantly reduces their size and introduces additional optimization that does not affect the accuracy of the model. By all means, there are options that allow further reduction of the model to make the model as fast as possible, however, the accuracy of the model then declines (Abadi et al., 2015).

After a successful conversion, a new file was created whose implementation was then possible within the mobile application. After its implementation and some additional tweaking, the following result was achieved after selecting a pre-existing image from the user's gallery (see Figure 7).

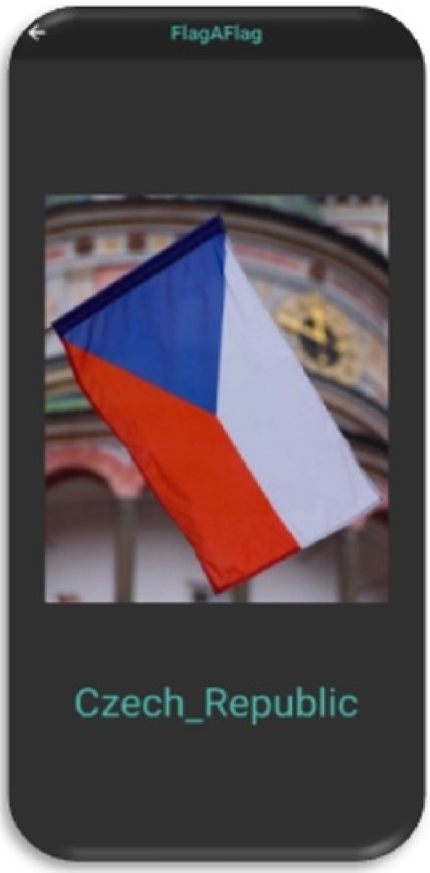


Figure 7. *Correct classification of the Czech flag*

Source: *Authors*

Upon further experimentation, some of the flags were incorrectly classified. One possible reason was the lack of similarity to the training data, that is, the model had previously not seen a similar flag image. In order to increase the accuracy of flag identification, i.e., classification, it was necessary to introduce a crop function as it enabled the reduction of unnecessary information inside the image, therefore increasing the model's accuracy when making predictions. The next subsection of this paper focuses on the implementation of the said function.

3.5.3. Implementation of the crop function

Images often come in all shapes and sizes and so it was necessary to anticipate that images would not be perfect, and because of this implementing a crop function was paramount. Although it is possible to crop an image within the gallery, developing a feature like this was crucial to make the application as fluid as possible. This way, if a chosen image contains two flags the user can select which one is going to be classified.

The difference in classification is visible in Figure 8. On the left side, a misclassification of the Hungarian flag is shown, whereas the right side depicts the correct classification after the use of the implemented crop function. Here it is obvious that reducing redundant information in the image increased the model accuracy since with only one flag being present in the image the correct flag could be identified. And with this, the process of developing a mobile application was completed.

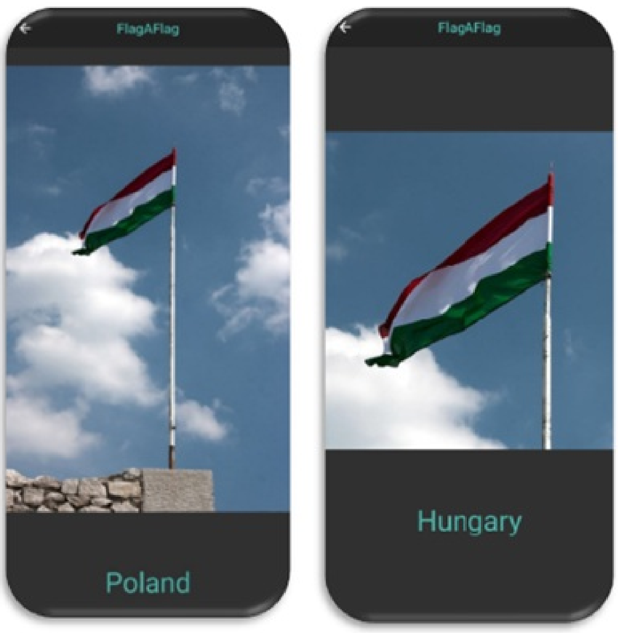


Figure 8. *Left: misclassification of the Hungarian flag, right: correct classification after the use of the implemented crop function*

Source: *Authors*

4. Evaluation, Result Discussion, and Future Work

To properly assess the functionality of the application “FlagAFlag” and evaluate the accuracy of the model, 118 images were manually and randomly selected in order to classify each flag at least two times. New images given to the application had previously not been seen by the model or the model had not been trained on them. Some of the flags were tested more than twice because of the issues with data similarity, which has been discussed in subsection 3.2. This way it was possible to check for possible problems and patterns that caused errors so that the model could be appropriately retuned and readapted. After testing, the results were surprising. Out of 118 flags, 113 (95.6%) of the flags were correctly classified. This figure was very close to the figure obtained during model validation and training; however, it was necessary to further investigate which flags were not classified correctly and for what reasons.

Comparing the results of this paper to the ones presented in section 2, it is clear that the accuracy is quite similar. It is, however, difficult to make a clear comparison between the different research due to different datasets and number of classes present in the model. It is also important to note that flag similarity plays a big role in the correct classification of a flag. For this reason, it is logical that a model with more classes will have more similar flags and will therefore possibly have worse accuracy. For accuracy to be as high as possible, the quality of the dataset must be closely monitored.

Taking an image like the one shown in Figure 9 as an example; this image belongs to the Netherlands, however, the flag itself is more similar to Luxembourg because of the light blue color present within the flag. This is problematic as the shade of blue present in the last stripe of the flag is the only way one can tell the difference between these two flags. Because of the light present in the image, the color blue appears to be slightly lighter and for that reason, the selected image is misclassified. These two countries, alongside France, posed the biggest obstacle during the training and testing of the model. By rotating the flag of the Netherlands, it becomes increasingly difficult to notice the differences between France and Luxembourg. In this example, even for the human eye, it is difficult to correctly assess which flag is shown on the image.

A possible solution to this problem would be to increase the collection of images or to significantly readapt the model, however, even with further modifications done to the model, the identification of these three countries would remain problematic. Another possible solution to this problem would actually be to reduce the size of the dataset. This would allow the model to learn from the data that was considered to be of the highest quality, however, this would not be reflective of the data gathered by its users as that data would be unpredictable and imperfect. Therefore, it is necessary to control the quality of the data on which the model is being trained on.

Alongside some hindrances, there were some surprising classifications. The very same flag of Slovenia shown in Figure 9 can be taken as an example. Although the coat of arms of Slovenia is visible to the human eye, due to its resemblance to Slavic flags, the authors expected that the application “FlagAFlag” would have misclassified this image. Nevertheless, the application successfully identified this flag and exceeded the initial expectations.

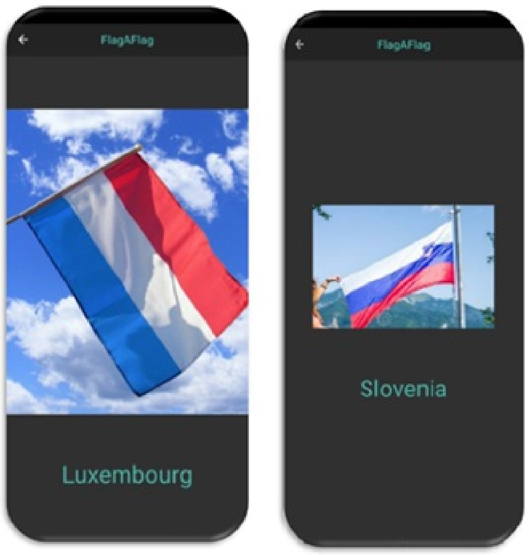


Figure 9. *Left: Misclassification of the flag of the Netherlands, right: correct classification of the Slovenian flag*

Source: *Authors*

4.1. Future development of the application "FlagAFlag"

For now, the "FlagAFlag" application is ready for experimental use in the real world, but it would be necessary to further improve the model and the mobile application itself. Despite the high accuracy, it is necessary to further expand the data collection to include more nations, i.e., flag classes so that the application could be tested on more data. This would provide more information about the steps necessary to tune the model in the right direction. In addition, further expansion of the collection would mean that the artificial neural network would have to be retuned and readjusted in order to accurately process the new dataset.

Within this mobile application, it is possible to add several additional features such as an option to save the classified image. Another one of the possible additions would be an option to share captured and classified images. Captured images of the flags could be shared with other users or sent to a server that collects all images classified by using the application. This way, the dataset could be further expanded, and each user would have the opportunity to contribute to the development of the mobile application. This could certainly improve the model and ensure greater accuracy in the future. Of course, images would have to be manually checked after classification to ensure the highest possible quality of data.

5. Conclusion

In this research, a functional mobile application called "FlagAFlag" was developed. It enables classification, i.e., identification of national flags. All the necessary steps to create such an application were presented in the paper and some problems were raised.

When it comes to the first research question, during the process of collecting the dataset, it was necessary to acquire images similar to those that users would end up feeding into the application. The process of data acquisition was automated via a Python program. Running this program resulted in a dataset that contained over 16,000 images. For the dataset to be reflective of the real-world data, during manual inspection the average number of images per class dropped from ca. 350 to ca. 40, as many of the images did not fit the stated criterion. The quality and quantity of the dataset used for training had a large impact on the quality of the model used for classification and its accuracy. To better the model, the dataset was expanded using a specific class provided by Keras, and two additional scripts for modifying the dataset were written using the Python programming language. This augmentation of the dataset improved the accuracy of the model and confirmed the main hypothesis of this research, i.e., that the position of the flag within the image affected the accuracy of the model, and hence, the efficacy of the developed application. Also, the augmentation of the dataset resulted in higher validation accuracy with minimal training loss accuracy.

Regarding the second research question in this paper, the tuning and adaptation of the artificial neural network were quite challenging as the whole development process took 45 hours in total. Since the objective of the mobile application was to classify images, Convolutional Neural Network was chosen as its architecture. In order for the model to display the probability of the input image, it was necessary to use a combination named Softmax loss as it allowed the network to output a probability over the classes found in the dataset for the selected and earlier-stored images or captured photographs. In order to combat overfitting, dropout layers were adopted, and this further increased the real-world accuracy of the model.

With the model having reached satisfying accuracy, the development of the mobile application was initiated. Some of the design issues were brought up, such as the need to create a minimalistic user interface. Another issue arose when trying to correctly classify the selected flag. The reason why this issue had appeared was due to unnecessary information present inside the image. It was solved with the implementation of the crop function, and at that point the application was ready for testing in real-world scenarios. This answers the third research question in this paper.

The testing was done on roughly 120 images the model had previously not seen before and its results were surprising with more than 95% correctly classified images depicting flags. Regarding the fourth research question, some classifications remained problematic due to the similarity of the colors and shapes present within the problematic flags. For this problem to be resolved and contained, it is necessary to control the quality of the data on which the model is being trained on. The paper also elaborated on the potential use of the mobile application “FlagAFlag” in the real world, as well as its future and possible updates that would improve its use and efficiency.

References

- Abadi, M. et al. (2015) “TensorFlow Lite guide”. https://www.tensorflow.org/lite/guide [Accessed 28 Apr. 2021]
- Appen (2018) “Great Machine Learning Data: It’s Not About Quantity or Quality”.https://appen.com/blog/great-machine-learning-data-quantity-or-quality/ [Accessed 28 Apr. 2021]
- Cowley, E. and Radich, Q. (2019) “What is a machine learning model?”.https://docs.microsoft.com/en-us/windows/ai/windows-ml/what-is-a-machine-learning-model [Accessed 28 Apr. 2021]
- Depositphotos (n.d.) “Ballot Box Croatian Flag Background”.https://depositphotos.com/326246184/stock-photo-ballot-box-croatian-flag-background.html [Accessed 28 Apr. 2021]
- Deng, J. et al. (2009) „ImageNet: A large-scale hierarchical image database“.*2009 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 248-255
- Dunđer, I. (2020) “Machine Translation System for the Industry Domain and Croatian Language”, Journal of Information and Organizational Sciences, 44(1), pp. 33-50
- Dunđer, I. and Pavlovski, M. (2019a) “Behind the Dystopian Sentiment: a Sentiment Analysis of George Orwell’s 1984”, 42nd International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO), pp. 577-582
- Dunđer, I. and Pavlovski, M. (2019b) “Through the Limits of Newspeak: an Analysis of the Vector Representation of Words in George Orwell’s 1984”, 42nd International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO), pp. 583-588
- Dunđer, I., Seljan, S. and Pavlovski, M. (2020) “Automatic Machine Translation of Poetry and a Low-Resource Language Pair”, 43rd International Convention on Information, Communication and Electronic Technology (MIPRO), pp. 1034-1039
- Dunđer, I., Seljan, S. and Stančić, H. (2015) “Koncept automatske klasifikacije registraturnoga i arhivskoga gradiva”, 48. savjetovanje hrvatskih arhivista (HAD), pp. 195-211
- Gómez, R. (2018) “Understanding Categorical Cross-Entropy Loss, Binary Cross-Entropy Loss, Softmax Loss, Logistic Loss, Focal Loss and all those confusing names”. https://gombru.github.io/2018/05/23/cross_entropy_loss/ [Accessed 28 Apr. 2021]
- Goodfellow, I., Bengio, Y. and Courville, A. (2016) “Deep Learning (Adaptive Computation and Machine Learning series)”. MIT Press, ISBN-13:978-0262035613
- Hart, E., Cha, S.-H. and Tappert, C. C. (2004) “Interactive Flag Identification Using Image Retrieval Techniques”. https://www.researchgate.net/publication/221296911_Interactive_Flag_Identification_Using_Image_Retrieval_Techniques [Accessed 28 Apr. 2021]
- Kingma, D. P. and Ba, J. (2015) “Adam: A Method for Stochastic Optimization”. semanticscholar.org/paper/Adam%3A-A-Method-for-Stochastic-Optimization-Kingma-Ba/a6cb366736791bcccc5c8639de5a8f9636bf87e8#citing-papers [Accessed 5 October 2021]
- Lee, K. (2017) “Identifying Patterns: Why Do Some Flags Look Similar?”.

https://blogs.proquest.com/culturegrams/identifying-patterns-why-do-some-flags-look-similar/ [Accessed 12 Apr. 2021]. [Accessed 7 May 2021]

- Lodh, A. and Parekh, R. (2016) “Computer Aided Identification of Flags using Color Features”, International Journal of Computer Applications, 149(11), pp. 1–7. https://www.ijcaonline.org/archives/volume149/number11/26038-2016911587 [Accessed 7 May 2021]
- Lugović, S. and Dunđer I. (2017) “Automatic Information Behaviour Recognition”, 40th International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO 2017), pp. 1429-1432
- Lugović, S., Dunđer, I. and Horvat, M. (2016) “Techniques and applications of emotion recognition in speech”, 39th International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO), pp. 1278-1283
- Mahamd, S., Siddika, S. and Sony, N. J. (2018) “National Flag Recognition Using CNN”. http://dspace.daffodilvarsity.edu.bd:8080/handle/123456789/2644 [Accessed 29 September 2021]
- Pelchat, M. and Gelowitz, C. M. (2020) "Neural Network Music Genre Classification", Canadian Journal of Electrical and Computer Engineering, vol. 43, no. 3, pp. 170-173. https://doi.org/10.1109/CJECE.2020.2970144 [Accessed 7 May 2021]
- Roh, Y., Heo, G. and Whang, S. E. (2019) “A Survey on Data Collection for Machine Learning: A Big Data - AI Integration Perspective”, IEEE Transactions on Knowledge and Data Engineering, 33(4), pp. 1328–1347. https://arxiv.org/pdf/1811.03402.pdf [Accessed 7 May 2021]
- Said, Y. and Barr, M. (2021) “Countries flags detection based on local context network and color features.” Multimed Tools Appl, 80, pp. 14753–14765. https://doi.org/10.1007/s11042-021-10509-8 [Accessed 7 May 2021]
- Sladojevic, S. et al. (2016) “Deep Neural Networks Based Recognition of Plant Diseases by Leaf Image Classification”, Computational Intelligence and Neuroscience, vol. 2016. https://doi.org/10.1155/2016/3289801 [Accessed 7 May 2021]
- Warden, P. (2017) “How many images do you need to train a neural network?”. https://petewarden.com/2017/12/14/how-many-images-do-you-need-to-train-a-neural-network/ [Accessed 28 Apr. 2021]
- William-Ellis, M. (n.d.) “Photo of the Croatian Flag in Dubrovnik, Croatia”. https://matthewwilliamsellis.photoshelter.com/image/I0000yVQUyoX.DKI [Accessed 28 Apr. 2021]
- Zhang, Y. (2010) “New Advances in Machine Learning“. InTech, DOI: 10.5772/225, ISBN: 978-953-307-034-6.