1. What do you think it occurred during this model development (training & evaluation)?

There are many ways it could have gone wrong, but since it was already implemented they thought it was good enough so the testing scores were probably good. So it is probably due to training (and testing) on images that were not representative to the ones that the actual ANPR cameras would take.

It is possible that the training was done with data in perfect conditions, taken with a good camera and non moving cars. So the model performance might be very good on the training and testing images, but not good for the Imperfect data from real conditions.

On the other hand, the training images might be imbalanced, so maybe there were a lot of license plates with "O" but not a lot of "D", so while the accuracy might be good, the model did not learn to recognize a "D".

The quality of the camera might also have impacted the results. The articles mention that the images were blurry and hard to read even for a human, so training a model to do it is very hard.

Those 3 are the most probable causes, but there are other possibilities like using the wrong scores to evaluate it, simply not testing, overfitting, not enough data, etc.

2. How would you fix this behavior? Please provide at least 2 options explaining their pros and drawbacks

- If the data is imbalanced, using the f1 weighted score for training and testing would help. This is a pretty simple solution and would help the model in recognizing the letters that don't have many samples. The downside is we might misclassify other letters more often. However, since confusing any letter is equally bad, it will be worthwhile as long as we can significantly reduce the rate of mistakes of the minorities even with a slight rise in mistakes elsewhere.
- If the training images are in perfect conditions, getting a better sample would be great, but it would be costly to get it. So a simpler solution that would not require great costs could be to add noise and blur to the images so that they would be a little more representative, but the resulting images would not be as representative so there would still be space for many errors.
- Finally, if the quality of the cameras is not good, preprocessing the images to make them higher quality or highlight important parts would be a very good solution. This could be done with another model to enhance the image quality by rescaling or removing noise. To highlight important things a filter could be applied, like using a kernel to only pick up vertical lines, which would help differentiate the D from the O. This solution would take more time to develop and it might not give the best results.

3. What do you think it will occur when running this AI in a different country with different plates formats? How would you ensure system accuracy?

Running the same model on other countries will probably fail, other countries might use a different font, color, length and other constraints. Additionally to the model failing, the processing pipeline might have specific constraints, for example, another country might have more letters and the current algorithm only searches for 4.

To ensure it works in other countries, the constraints would have to be lost, so search for letters and numbers all the time.

Getting images from other countries would be the best, so that the model is actually trained for other car plates. If this is not feasible, applying filters and transformations like noise, blur, changing colors, stretching the image...

4. Do you know any OCR (Optical Character Recognition) algorithms (Deep learning based) that could be used here?

Easy OCR can be finetuned for this specific case and I would also use opencv to preprocess the image and prepare it. So the image can be turned into black and white, I would test some kernel filters to try and help make the recognition perform better. Then finetune EasyOCR to detect the car plaques. And finally I'd build a pipeline to streamline the process.

I would also add a warning system for the cases where the model is not sure so that someone could verify the accuracy before giving a fine. This way, if the image is too blurry for the model, there's an extra layer of protection. For this, easyOCR can provide a confidence level, so if it is too low we can make the warning.

5. Explain a Computer Vision / Artificial Intelligence project in which you have participated (goals, your role, difficulties you found, how they were solved, ...)

I recently worked on a project to build some models to classify ships based on their sound and compare how they performed. I worked on preparing and processing the available data since it was unorganized. I made an analysis on the available data to see if there were any useful patterns or ways to enhance the model.Then I created different models to test out what was the best option. We used 3 types of models, the idea was to see how far we could get them, it was a CNN, a Logistic regression and a random forest.

The hardest thing was to choose what to feed the logistic regression and random forest, since you can't really give them a wav file. To solve this I opted to pull a lot of different calculations, since they were fairly easy to implement, like the mel coefficients and tested them all out to see how they worked.

Organizing the data was also a problem, since the metadata did not include a direct reference to the matching audios. There was no standard naming convention but they did have some data that I could match on the file names. It resulted in a pretty big algorithm to find if there was something to match by testing different possibilities.