Bird Recognition in the City of Peacetopia (Case Study)

Problem Statement

This example is adapted from a real production application, but with details disguised to protect confidentiality.

You are a famous researcher in the City of Peacetopia. The people of Peacetopia have a common characteristic: they are afraid of birds. To save them, you have to build an algorithm that will detect any bird flying over Peacetopia and alert the population.

The City Council gives you a dataset of 10,000,000 images of the sky above Peacetopia, taken from the city's security cameras. They are labelled:

- y = 0: There is no bird on the image
- y = 1: There is a bird on the image

Your goal is to build an algorithm able to classify new images taken by security cameras from Peacetopia.

There are a lot of decisions to make:

- What is the evaluation metric?
- How do you structure your data into train/dev/test sets?

Metric of success

The City Council tells you that they want an algorithm that

1. Has high accuracy
2. Runs quickly and takes only a short time to classify a new image.
3. Can fit in a small amount of memory, so that it can run in a small processor that the city will attach to many different security cameras.

Note: **Having three evaluation metrics makes it harder for you to quickly choose between two different algorithms, and will slow down the speed with which your team can iterate. True/False?**

True

False

After further discussions, the city narrows down its criteria to:

- "We need an algorithm that can let us know a bird is flying over Peacetopia as accurately as possible."
- "We want the trained model to take no more than 10sec to classify a new image."
- "We want the model to fit in 10MB of memory."

If you had the three following models, which one would you choose?

Test Accuracy	**Runtime**	**Memory size**
98%	**9 sec**	**9MB**

Test Accuracy	Runtime	Memory size
97%	1 sec	3MB

Test Accuracy	Runtime	Memory size
99%	13 sec	9MB

Test Accuracy	Runtime	Memory size
97%	3 sec	2MB

Based on the city's requests, which of the following would you say is true?

Accuracy is a satisficing metric; running time and memory size are an optimizing metric.

Accuracy, running time and memory size are all satisficing metrics because you have to do sufficiently well on all three for your system to be acceptable.

Accuracy, running time and memory size are all optimizing metrics because you want to do well on all three.

Accuracy is an optimizing metric; running time and memory size are a satisficing metrics.

Structuring your data

Before implementing your algorithm, you need to split your data into train/dev/test sets. Which of these do you think is the best choice?

Train	Dev	Test
3,333,334	3,333,333	3,333,333

Train	Dev	Test
6,000,000	1,000,000	3,000,000

Train	Dev	Test
6,000,000	3,000,000	1,000,000

Train	**Dev**	**Test**
9,500,000	**250,000**	**250,000**

After setting up your train/dev/test sets, the City Council comes across another 1,000,000 images, called the "citizens' data". Apparently the citizens of Peacetopia are so scared of birds that they volunteered to take pictures of the sky and label them, thus contributing these additional 1,000,000 images. These images are different from the distribution of images the City Council had originally given you, but you think it could help your algorithm.

Notice that adding this additional data to the training set will make the distribution of the training set different from the distributions of the dev and test sets.

Is the following statement true or false?

"You should not add the citizens' data to the training set, because if the training distribution is different from the dev and test sets, then this will not allow the model to perform well on the test set."

True

False

One member of the City Council knows a little about machine learning, and thinks you should add the 1,000,000 citizens' data images to the test set. You object because:

A bigger test set will slow down the speed of iterating because of the computational expense of evaluating models on the test set.

The 1,000,000 citizens' data images do not have a consistent x-->y mapping as the rest of the data (similar to the New York City/Detroit housing prices example from lecture).

The test set no longer reflects the distribution of data (security cameras) you most care about.

This would cause the dev and test set distributions to become different. This is a bad idea because you're not aiming where you want to hit.

You train a system, and its errors are as follows (error = 100%-Accuracy):

Training set error	4.0%
Dev set error	4.5%

This suggests that one good avenue for improving performance is to train a bigger network so as to drive down the 4.0% training error. Do you agree?

Yes, because having 4.0% training error shows you have a high bias.

Yes, because this shows your bias is higher than your variance.

No, because there is insufficient information to tell.

No, because this shows your variance is higher than your bias.

You ask a few people to label the dataset so as to find out what is human-level performance. You

find the following levels of accuracy:

Bird watching expert #1	0.3% error
Bird watching expert #2	0.5% error
Normal person #1 (not a bird watching expert)	1.0% error
Normal person #2 (not a bird watching expert)	1.2% error

If your goal is to have "human-level performance" be a proxy (or estimate) for Bayes error, how would you define "human-level performance"?

0.4% (average of 0.3 and 0.5)

0.0% (because it is impossible to do better than this)

0.3% (accuracy of expert #1)

0.75% (average of all four numbers above)

Which of the following statements do you agree with?

A learning algorithm's performance can never be better than human-level performance nor better than Bayes error.

A learning algorithm's performance can be better than human-level performance but it can never be better than Bayes error.

A learning algorithm's performance can never be better than human-level performance but it can be better than Bayes error.

A learning algorithm's performance can be better than human-level performance and better than Bayes error.

You find that a team of ornithologists debating and discussing an image gets an even better 0.1% performance, so you define that as "human-level performance." After working further on your algorithm, you end up with the following:

Human-level performance	0.1%
Training set error	2.0%

Dev set error	2.1%

Based on the evidence you have, which two of the following four options seem the most promising to try? (Check two options.)

Try increasing regularization.

Get a bigger training set to reduce variance.

Train a bigger model to try to do better on the training set.

Try decreasing regularization.

You also evaluate your model on the test set, and find the following:

Human-level performance	0.1%
Training set error	2.0%
Dev set error	2.1%
Test set error	7.0%

What does this mean? (Check the two best options.)

You should get a bigger test set.

You have overfit to the dev set.

You have underfit to the dev set.

You should try to get a bigger dev set.

After working on this project for a year, you finally achieve:

Human-level performance	0.10%

Training set error	0.05%
Dev set error	0.05%

What can you conclude? (Check all that apply.)

It is now harder to measure avoidable bias, thus progress will be slower going forward.

If the test set is big enough for the 0.05% error estimate to be accurate, this implies Bayes error is ≤ 0.05

With only 0.09% further progress to make, you should quickly be able to close the remaining gap to 0%

This is a statistical anomaly (or must be the result of statistical noise) since it should not be possible to surpass human-level performance.

It turns out Peacetopia has hired one of your competitors to build a system as well. Your system and your competitor both deliver systems with about the same running time and memory size. However, your system has higher accuracy! However, when Peacetopia tries out your and your competitor's systems, they conclude they actually like your competitor's system better, because even though you have higher overall accuracy, you have more false negatives (failing to raise an alarm when a bird is in the air). **What should you do?**

Rethink the appropriate metric for this task, and ask your team to tune to the new metric.

Look at all the models you've developed during the development process and find the one with the lowest false negative error rate.

Pick false negative rate as the new metric, and use this new metric to drive all further development.

Ask your team to take into account both accuracy and false negative rate during development.

You've handily beaten your competitor, and your system is now deployed in Peacetopia and is protecting the citizens from birds! But over the last few months, a new species of bird has been slowly migrating into the area, so the performance of your system slowly degrades because your data is being tested on a new type of data.

You have only 1,000 images of the new species of bird. The city expects a better system from you within the next 3 months. Which of these should you do first?

Add the 1,000 images into your dataset and reshuffle into a new train/dev/test split.

Put the 1,000 images into the training set so as to try to do better on these birds.

Use the data you have to define a new evaluation metric (using a new dev/test set) taking into account the new species, and use that to drive further progress for your team.

Try data augmentation/data synthesis to get more images of the new type of bird.

The City Council thinks that having more Cats in the city would help scare off birds. They are so happy with your work on the Bird detector that they also hire you to build a Cat detector. (Wow Cat detectors are just incredibly useful aren't they.) Because of years of working on Cat detectors, you have such a huge dataset of 100,000,000 cat images that training on this data takes about two weeks. **Which of the statements do you agree with? (Check all that agree.)**

Buying faster computers could speed up your teams' iteration speed and thus your team's productivity.

If 100,000,000 examples is enough to build a good enough Cat detector, you might be better off training with just 10,000,000 examples to gain a ≈10x improvement in how quickly you can run experiments, even if each model performs a bit worse because it's trained on less

data.

Needing two weeks to train will limit the speed at which you can iterate.

Having built a good Bird detector, you should be able to take the same model and hyperparameters and just apply it to the Cat dataset, so there is no need to iterate.

Autonomous Driving (Case Study)

To help you practice strategies for machine learning, in this week we'll present another scenario and ask how you would act. We think this "simulator" of working in a machine learning project will give a task of what leading a machine learning project could be like!

You are employed by a startup building self-driving cars. You are in charge of detecting road signs (stop sign, pedestrian crossing sign, construction ahead sign) and traffic signals (red and green lights) in images. The goal is to recognize which of these objects appear in each image. As an example, the above image contains a pedestrian crossing sign and red traffic lights

$$y^{(i)} = \begin{bmatrix} 0 \\ 1 \\ 0 \\ 1 \\ 0 \end{bmatrix} \begin{matrix} \text{“stop sign”} \\ \text{“pedestrian crossing sign”} \\ \text{“construction ahead sign”} \\ \text{“red traffic light”} \\ \text{“green traffic light”} \end{matrix}$$

Your 100,000 labeled images are taken using the front-facing camera of your car. This is also the distribution of data you care most about doing well on. You think you might be able to get a much larger dataset off the internet, that could be helpful for training even if the distribution of internet data is not the same.

You are just getting started on this project. What is the first thing you do? Assume each of the steps below would take about an equal amount of time (a few days).

Spend a few days collecting more data using the front-facing camera of your car, to better understand how much data per unit time you can collect.

Spend a few days checking what is human-level performance for these tasks so that you can get an accurate estimate of Bayes error.

Spend a few days training a basic model and see what mistakes it makes.

Spend a few days getting the internet data, so that you understand better what data is available.

Your goal is to detect road signs (stop sign, pedestrian crossing sign, construction ahead sign) and traffic signals (red and green lights) in images. The goal is to recognize which of these objects appear in each image. You plan to use a deep neural network with ReLU units in the hidden layers.

For the output layer, a softmax activation would be a good choice for the output layer because this is a multi-task learning problem. True/False?

True

False

You are carrying out error analysis and counting up what errors the algorithm makes. Which of these datasets do you think you should manually go through and carefully examine, one image at a time?

500 randomly chosen images

10,000 randomly chosen images

500 images on which the algorithm made a mistake

10,000 images on which the algorithm made a mistake

After working on the data for several weeks, your team ends up with the following data:

- 100,000 labeled images taken using the front-facing camera of your car.
- 900,000 labeled images of roads downloaded from the internet.
- Each image's labels precisely indicate the presence of any specific road signs and traffic signals or combinations of them. For example, y(i) = [1 0 0 1 0] means the image contains a stop sign and a red traffic light.

Because this is a multi-task learning problem, you need to have all your y(i) vectors fully labeled. If one example is equal to [0 ? 1 1 ?] then the learning algorithm will not be able to use that example. True/False?

False

True

The distribution of data you care about contains images from your car's front-facing camera; which comes from a different distribution than the images you were able to find and download off the internet. How should you split the dataset into train/dev/test sets?

Mix all the 100,000 images with the 900,000 images you found online. Shuffle everything. Split the 1,000,000 images dataset into 980,000 for the training set, 10,000 for the dev set and 10,000 for the test set.

Mix all the 100,000 images with the 900,000 images you found online. Shuffle everything. Split the 1,000,000 images dataset into 600,000 for the training set, 200,000 for the dev set and 200,000 for the test set.

Choose the training set to be the 900,000 images from the internet along with 80,000 images from your car's front-facing camera. The 20,000 remaining images will be split equally in dev and test sets.

Choose the training set to be the 900,000 images from the internet along with 20,000 images from your car's front-facing camera. The 80,000 remaining images will be split equally in dev and test sets.

Assume you've finally chosen the following split between of the data:

Dataset:	Contains:	Error of the algorithm:
Training	940,000 images randomly picked from (900,000 internet images + 60,000 car's front-facing camera images)	8.8%
Training -Dev	20,000 images randomly picked from (900,000 internet images + 60,000 car's front-facing camera images)	9.1%
Dev	20,000 images from your car's front-facing camera	14.3%
Test	20,000 images from the car's front-facing camera	14.8%

You also know that human-level error on the road sign and traffic signals classification task is around 0.5%. Which of the following are True? (Check all that apply).

You have a large variance problem because your training error is quite higher than the human-level error.

You have a large avoidable-bias problem because your training error is quite a bit higher than the human-level error.

You have a large data-mismatch problem because your model does a lot better on the training-dev set than on the dev set

Your algorithm overfits the dev set because the error of the dev and test sets are very close.

You have a large variance problem because your model is not generalizing well to data from the same training distribution but that it has never seen before.

Based on the table from the previous question, a friend thinks that the training data distribution is much easier than the dev/test distribution. What do you think?

Your friend is right. (I.e., Bayes error for the training data distribution is probably lower than for the dev/test distribution.)

There's insufficient information to tell if your friend is right or wrong.

Your friend is wrong. (I.e., Bayes error for the training data distribution is probably higher than for the dev/test distribution.)

You decide to focus on the dev set and check by hand what are the errors due to. Here is a table summarizing your discoveries:

Overall dev set error	15.3%
Errors due to incorrectly labeled data	4.1%
Errors due to foggy pictures	8.0%
Errors due to rain drops stuck on your car's front-facing camera	2.2%
Errors due to other causes	1.0%

In this table, 4.1%, 8.0%, etc. are a fraction of the total dev set (not just examples your algorithm mislabeled). For example, about 8.0/15.3 = 52% of your errors are due to foggy pictures.

The results from this analysis implies that the team's highest priority should be to bring more foggy pictures into the training set so as to address the 8.0% of errors in that category. True/False?

Additional Note: there are subtle concepts to consider with this question, and you may find arguments for why some answers are also correct or incorrect. We recommend that you spend time reading the feedback for this quiz, to understand what issues that you will want to consider when you are building your own machine learning project.

First start with the sources of error that are least costly to fix.

False because it depends on how easy it is to add foggy data. If foggy data is very hard and costly to collect, it might not be worth the team's effort.

True because it is greater than the other error categories added together (8.0 > 4.1+2.2+1.0).

True because it is the largest category of errors. We should always prioritize the largest category of error as this will make the best use of the team's time.

You can buy a specially designed windshield wiper that help wipe off some of the raindrops on the front-facing camera. Based on the table from the previous question, which of the following statements do you agree with?

2.2% would be a reasonable estimate of how much this windshield wiper will improve performance.

2.2% would be a reasonable estimate of how much this windshield wiper could worsen performance in the worst case.

2.2% would be a reasonable estimate of the minimum amount this windshield wiper could improve performance.

2.2% would be a reasonable estimate of the maximum amount this windshield wiper could improve performance.

You decide to use data augmentation to address foggy images. You find 1,000 pictures of fog off the internet, and "add" them to clean images to synthesize foggy days, like this:

image from front-facing camera

+

foggy image from the internet

=

synthesized foggy image

Which of the following statements do you agree with?

Adding synthesized images that look like real foggy pictures taken from the front-facing camera of your car to training dataset won't help the model improve because it will introduce avoidable-bias.

So long as the synthesized fog looks realistic to the human eye, you can be confident that the synthesized data is accurately capturing the distribution of real foggy images (or a subset of it), since human vision is very accurate for the problem you're solving.

There is little risk of overfitting to the 1,000 pictures of fog so long as you are combining it with a much larger (>>1,000) of clean/non-foggy images.

After working further on the problem, you've decided to correct the incorrectly labeled data on the dev set. Which of these statements do you agree with? (Check all that apply).

You should not correct the incorrectly labeled data in the test set, so that the dev and test sets continue to come from the same distribution.

You should correct incorrectly labeled data in the training set as well so as to avoid your training set now being even more different from your dev set.

You should also correct the incorrectly labeled data in the test set, so that the dev and test sets continue to come from the same distribution.

You do not necessarily need to fix the incorrectly labeled data in the training set, because it's okay for the training set distribution to differ from the dev and test sets. Note that it is important that the dev set and test set have the same distribution.

So far your algorithm only recognizes red and green traffic lights. One of your colleagues in the startup is starting to work on recognizing a yellow traffic light. (Some countries call it an orange light rather than a yellow light; we'll use the US convention of calling it yellow.) Images containing yellow lights are quite rare, and she doesn't have enough data to build a good model. She hopes you can help her out using transfer learning.

What do you tell your colleague?

She should try using weights pre-trained on your dataset, and fine-tuning further with the yellow-light dataset.

Recommend that she try multi-task learning instead of transfer learning using all the data.

You cannot help her because the distribution of data you have is different from hers, and is also lacking the yellow

label.

If she has (say) 10,000 images of yellow lights, randomly sample 10,000 images from your dataset and put your and her data together. This prevents your dataset from “swamping” the yellow lights dataset.

Another colleague wants to use microphones placed outside the car to better hear if there are other vehicles around you. For example, if there is a police vehicle behind you, you would be able to hear their siren. However, they don’t have much to train this audio system. How can you help?

Either transfer learning or multi-task learning could help our colleague get going faster.

Multi-task learning from your vision dataset could help your colleague get going faster. Transfer learning seems significantly less promising.

Transfer learning from your vision dataset could help your colleague get going faster. Multi-task learning seems significantly less promising.

Neither transfer learning nor multi-task learning seems promising.

To recognize red and green lights, you have been using this approach:

- (A) Input an image (x) to a neural network and have it directly learn a mapping to make a prediction as to whether there’s a red light and/or green light (y).

A teammate proposes a different, two-step approach:

- (B) In this two-step approach, you would first (i) detect the traffic light in the image (if any), then (ii) determine the color of the illuminated lamp in the traffic light.

Between these two, Approach B is more of an end-to-end approach because it has distinct steps for the input end and the output end. True/False?

False

True

Approach A (in the question above) tends to be more promising than approach B if you have a _____ (fill in the blank).

Multi-task learning problem.

Large bias problem.

Large training set

Problem with a high Bayes error.