Carrying out error analysis

Imagine your cat classifier has 90% accuracy and 10% error. From that 10% error you think it's due to wrongly classifying some dogs as cats. Should you try to make your cat classifier do better on dogs? To answer this question we must do error analysis. To achieve this analysis get 100 images on the dev set that are being mislabeled. Count up how many are dogs. Imagine for example that only 5/100 images are dogs. That means that even if you've fully solved the dog classification problem your error would only go down 5%, coming from 10% to 9.5% at best. This is the "ceiling" of your next step.

Multiple ideas can be evaluated in parallel to quickly decide which one might have bigger impact on your error reduction process.

Cleaning up incorrectly labeled data

DL algorithms are quite robust to random errors but much less robust to systematic errors. The incorrectly labeled data impact can be analyzed as any other error analysis metric with the same estimation on the resulting impact.

Build your first system quickly, then iterate

Training and testing on different distributions

Deep learning algorithms are hunger for data and because of that teams sometimes just feed data to the algorithms without checking if the distribution of the train/test/dev sets are compatible with their objectives. The train set should have as much data as possible that helps the algorithm learn in the right direction. But the dev and test set are important to determine if the algorithm is really learning what is important to the product or not.

In a given example there are two main sets of images:

1. high quality cat pictures downloaded from the web with well framed cats (~200k pics)
2. low quality cat pictures taken from users using your app with low resolution, unfocused, cats not well framed, etc. (~10k pics)

If your app objective is to work well on the number 2 but the amount of data of 1 is much higher how should you proceed?

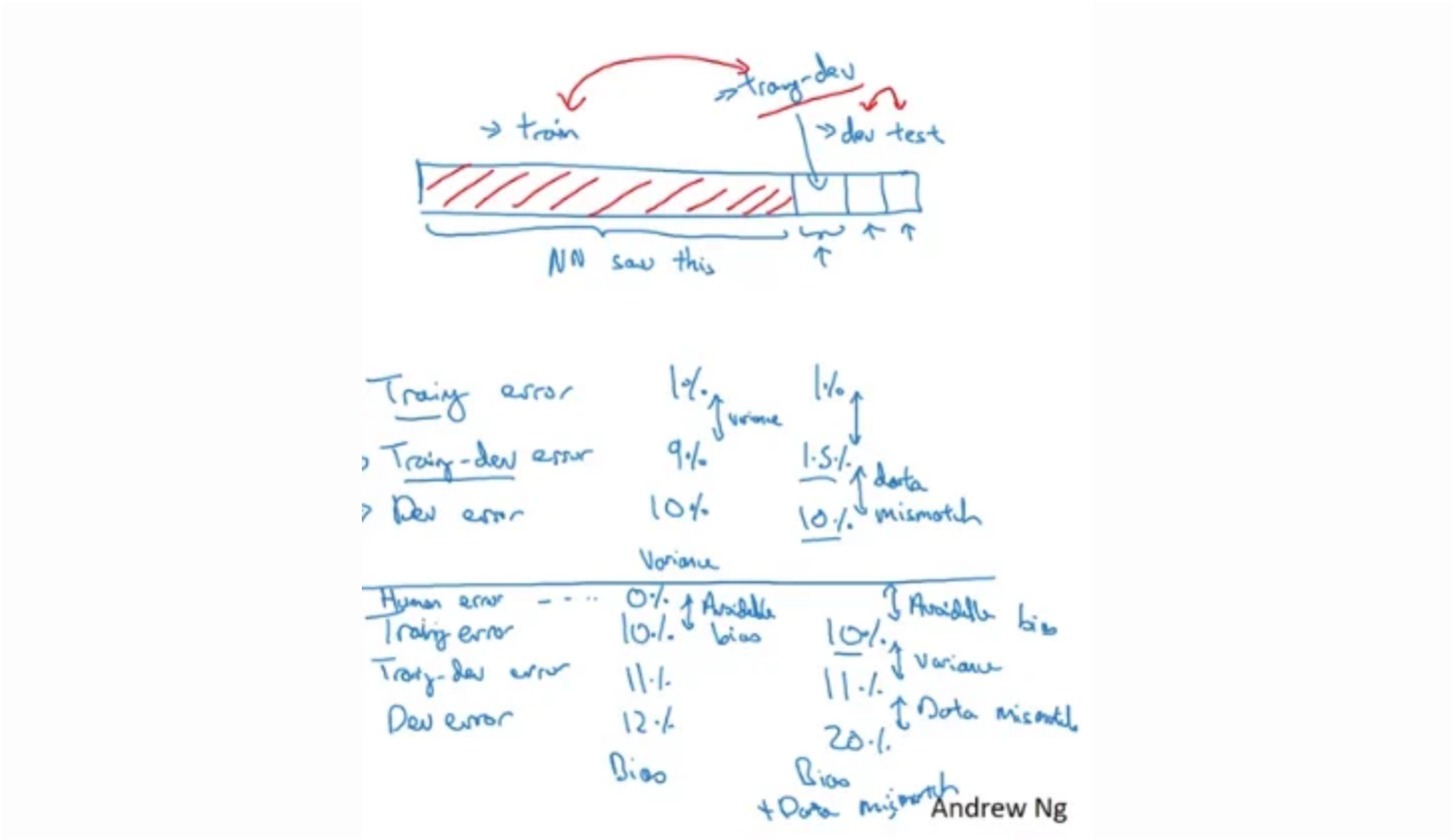
One option is to mix both 1 and 2 equally ending up with 210k images set to be divided into training, dev and test. The advantage is that all the sets will have the same distribution but the main disadvantage is that since your test and dev sets have a higher probability of having images from the group 1 that is the main group being optimized instead of the group 2, which is your real data.

A second option is instead of mixing it all together using all the images from 1 + half of the images from 2 into the training set. That way you'll have at least some of the real images you're trying top optimize in the training set. If you take the remaining 5k images and divide them into dev and test set your metrics on the validation and test set will help your algorithm aim much closer to the distribution you want and also provide extra learning from the other images in the group 1. This strategy maker the distribution is different across the sets but that isn't a problem since you're really aiming for the problem you want and also providing some of your real data into the training set.

Bias and Variance with mismatched data distributions

If you're training on different distributions set but want to really understand how much of the error in your dev set comes from the variation in distribution and how much comes from the variance you can create another dev set that comes from the same distribution as the training data so you can assess how much each one influences out.

In the image below we can check on 4 possible scenarios of errors. For reference the human error performance reference is ~ 0%



1. Scenario 1:
   - Training Error: 1%
   - Training-dev Error: 9%
   - Dev Error: 10%
   - As the difference from the training error and the traning-dev error is great we can point the there's a variance problem in the model where it cannot generalize well not even in the expected data (training-dev set which has the same distribution)
2. Scenario 2:
   - Training Error: 1%
   - Training-dev Error: 1.5%
   - Dev Error: 10%
   - Now the difference of the training-dev and the dev set is much more acute pointing out that the algorithm is performing well on the data is using to learn but that isn't really helping him on the data that matters. Sometimes referred as **data mismatch** problem
3. Scenario 3:
   - Training Error: 10%
   - Training-dev Error: 11%

- Dev Error: 12%
- Here we can check that the biggest problem is that the network is not even close enough to the error reference (~ 0%) indicating a bias problem

4. Scenario 4:
   - Training Error: 10%
   - Training-dev Error: 11%
   - Dev Error: 20%
   - At last multiple problems can raise at the same time. Here we see both data mismatch problem as well as bias problem.

	General Problem	Your specific problem
Human Level	“Human Level” 4%	6%
Error on actual trained on	“Training error” 7%	6%
Error on examples not trained on	“Training-dev error” 10%	“dev error” 6%

The errors across human level and error on actual trained are the avoidable bias. We can see in the table above that the avoidable bias in the specific problem was 0 indicating that the algorithm performed as well as humans would do. The difference between the error on the data actually trained on and not trained on is the variance which is also 0 in the specific problem. Finally the difference between the general approach and the specific approach errors on the data it wasn’t trained on is the data mismatch.

Addressing data mismatch

Data mismatch can be addressed by manual error analysis comparing the data on both differs. After this you can add more training data closer to your actual distribution. The insights part is tricky because it could become easily clear why the data differs as well it could be pretty hard or tricky.

Artificial data synthesis is a solution that might boost the performance of the system. But it’s possible that it might be a subset of examples in the world of examples available and turn the learning to an overfitting.

Transfer learning

One of the most powerful ideas in deep learning is that sometimes you can take knowledge the neural network has learned from one task and apply that knowledge to a separate task. You could have the neural network learn to recognize objects like cats and then use that knowledge or use part of that knowledge to help you do a better job reading x-ray scans. This is called transfer learning.

You could simply delete the n-last layers and re-insert your desired architecture to fit well on your problem. As for the trained part you have the option to use it frozen as it is and not inserting it in the new learning part or you could still allow the gradients take place and modify the errors.

As a guideline to how many layers remove/add and weather you should let the layers learn or not it’s good to use your available data size. If you have lots of data you could easily add more layers and let them learn since it’s unlikely to overfit. If you have just a small amount of data adding too many layers could easily overfit.

In transfer learning usually we want the model to learn faster or with less data.

Multi-task learning

So whereas in transfer learning, you have a sequential process where you learn from task A and then transfer that to task B. In multi-task learning, you start off simultaneously, trying to have one neural network do several things at the same time. And then each of these task helps hopefully all of the other task.

Multi-task learning mostly changes the output of the network, carrying out multiple classifications at time. To do this the loss function must be tweaked to perform summation not only over the first set of examples but as well as in all the desired classificators in the output. As we are tweaking the loss function it's also possible to deal with missing data on the carrying out of the derivatives. To deal with it simply sum over the existing marked values.

So when does multi-task learning make sense?

1. Training on a set of tasks that could benefit from having shared low-level features.
2. Amount of data you have for each task is quite similar
3. Can train a big enough network to do well on all the tasks

What is end-to-end deep learning?

Briefly, there have been some data processing systems, or learning systems that require multiple stages of processing. And what end-to-end deep learning does, is it can take all those multiple stages, and replace it usually with just a single neural network.

Whether to use end-to-end deep learning

Pros:

- Let the data speak
- Less hand-designing of componenets needed

Cons:

- May need large amount of data
- Excludes potentially useful hand-design components

Key question: Do you have sufficient data to learn a function of the complexity needed to map x to y? Here complexity is a subjective concept which nowadays can only be grasped by experience and there's no well defined way of doing this.

Quiz

1.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

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 getting the internet data, so that you understand better what data is available.
- ☑ Spend a few days training a basic model and see what mistakes it makes.
- ☐ 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 collecting more data using the front-facing camera of your car, to better understand how much data per unit time you can collect.

2.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

3.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?

- ☐ 10,000 images on which the algorithm made a mistake
- ☐ 10,000 randomly chosen images
- ☐ 500 randomly chosen images
- ☑ 500 images on which the algorithm made a mistake

4.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].T 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,?].T then the learning algorithm will not be able to use that example. True/False?

- ☐ True
- ☑ False

5.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?

- ☐ 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.
- ☐ 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.

- ☐ 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.
- ☑ 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.

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

7.Based on 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.)
- ☐ Your friend is wrong. (I.e., Bayes error for the training data distribution is probably higher than for the dev/test distribution.)
- ☑ There's insufficient information to tell if your friend is right or wrong.

8.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 14.3% 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). I.e. about 8.0/14.3 = 56% 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?

- ☐ True because it is the largest category of errors. As discussed in lecture, we should prioritize the largest category of error to avoid wasting the team's time.
- ☐ True because it is greater than the other error categories added together (8.0 > 4.1+2.2+1.0).
- ☑ False because this would depend on how easy it is to add this data and how much you think your team thinks it'll help.
- ☐ False because data augmentation (synthesizing foggy images by clean/non-foggy images) is more efficient.

9.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?

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

10.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: Which of the following statements do you agree with?

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

11.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).

- [x] 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 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 not correct the incorrectly labeled data in the test set, so that the dev and test sets continue to come from the same distribution
- [x] You should not correct incorrectly labeled data in the training set as it does not worth the time.

12.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?

- [x] She should try using weights pre-trained on your dataset, and fine-tuning further with the yellow-light dataset.
- [ ] 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.
- [ ] You cannot help her because the distribution of data you have is different from hers, and is also lacking the yellow label.
- [ ] Recommend that she try multi-task learning instead of transfer learning using all the data.

13.Another colleague wants to use microphones placed outside the car to better hear if there're 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?

- ☐ Transfer learning from your vision dataset could help your colleague get going faster. Multi-task learning seems significantly less promising.
- ☐ Multi-task learning from your vision dataset could help your colleague get going faster. Transfer learning seems significantly less promising.
- ☐ Either transfer learning or multi-task learning could help our colleague get going faster.
- ☑ Neither transfer learning nor multi-task learning seems promising.

14.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?

- ☐ True
- ☑ False

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

- ☑ Large training set
- ☐ Multi-task learning problem.
- ☐ Large bias problem.
- ☐ Problem with a high Bayes error.