Carrying out error analysis

(DESCRIPTION)
Text, Error Analysis. Carrying out error analysis.

(SPEECH)
Hello, and welcome back.

If you're trying to get a learning algorithm to do a task that humans can do.

And if your learning algorithm is not yet at the performance of a human.

Then manually examining mistakes that your algorithm is making, can give you insights into what to do next.

This process is called error analysis.

Let's start with an

(DESCRIPTION)
New slide, Look at dev examples to evaluate ideas.

(SPEECH)
example.

Let's say you're working on your cat classifier, and you've achieved 90% accuracy, or equivalently 10% error, on your dev set.

And let's say this is much worse than you're hoping to

(DESCRIPTION)
Two photos of small dogs appear.

(SPEECH)
do.

Maybe one of your teammates looks at some of the examples that the algorithm is misclassifying, and notices that it is miscategorizing some dogs as cats.

And if you look at these two dogs, maybe they look a little bit like a cat, at least at first glance.

So maybe your teammate comes to you with a proposal for how to make the algorithm do better, specifically on dogs, right?

You can imagine building a focus effort, maybe to collect more dog pictures, or maybe to design features specific to dogs, or something.

In order to make your cat classifier do better on dogs, so it stops misrecognizing these dogs as cats.

So the question is, should you go ahead and start a project focus on the dog problem?

There could be several months of works you could do in order to make your algorithm make few mistakes on dog pictures.

So is that worth your effort?

Well, rather than spending a few months doing this, only to risk finding out at the end that it wasn't that helpful.

Here's an error analysis procedure that can let you very quickly tell whether or not this could be worth your effort.

Here's what I recommend you do.

First, get about, say 100 mislabeled dev set examples, then examine them manually.

Just count them up one at a time, to see how many of these mislabeled examples in your dev set are actually pictures of dogs.

Now, suppose that it turns out that 5% of your 100 mislabeled dev set examples are pictures of dogs.

So, that is, if 5 out of 100 of these mislabeled dev set examples are dogs, what this means is that of the 100 examples.

Of a typical set of 100 examples you're getting wrong, even if you completely solve the dog problem, you only get 5 out of 100 more correct.

Or in other words, if only 5% of your errors are dog pictures, then the best you could easily hope to do, if you spend a lot of time on the dog problem.

Is that your error might go down from 10% error, down to 9.5% error, right?

So this a 5% relative decrease in error, from 10% down to 9.5%.

And so you might reasonably decide that this is not the best use of your time.

Or maybe it is, but at least this gives you a ceiling, right?

Upper bound on how much you could improve performance by working on the dog problem, right?

In machine learning, sometimes we call this the ceiling on performance.

Which just means, what's in the best case?

How well could working on the dog problem help you?

But now, suppose something else happens.

Suppose that we look at your 100 mislabeled dev set examples, you find that 50 of them are actually dog images.

So 50% of them are dog pictures.

Now you could be much more optimistic about spending time on the dog problem.

In this case, if you actually solve the dog problem, your error would go down from this 10%, down to potentially 5% error.

And you might decide that halving your error could be worth a lot of effort.

Focus on reducing the problem of mislabeled dogs.

I know that in machine learning, sometimes we speak disparagingly of hand engineering things, or using too much value insight.

But if you're building applied systems, then this simple counting procedure, error analysis, can save you a lot of time.

In terms of deciding what's the most important, or what's the most promising direction to focus on.

In fact, if you're looking at 100 mislabeled dev set examples, maybe this is a 5 to 10 minute effort.

To manually go through 100 examples, and manually count up how many of them are dogs.

And depending on the outcome, whether there's more like 5%, or 50%, or something else.

This, in just 5 to 10 minutes, gives you an estimate of how worthwhile this direction is.

And could help you make a much better decision, whether or not to spend the next few months focused on trying to find solutions to solve the problem of mislabeled dogs.

In this slide, we'll describe using error analysis to evaluate whether or not a single idea, dogs in this case, is worth working on.

(DESCRIPTION)
New slide, Evaluate multiple ideas in parallel.

(SPEECH)
Sometimes you can also evaluate multiple ideas in parallel doing error analysis.

For example, let's say you have several ideas in improving your cat detector.

Maybe you can improve performance on dogs?

Or maybe you notice that sometimes, what are called great cats, such as lions, panthers, cheetahs, and so on.

That they are being recognized as small cats, or house cats.

So you could maybe find a way to work on that.

Or maybe you find that some of your images are blurry, and it would be nice if you could design something that just works better on blurry images.

And maybe you have some ideas on how to do that.

(DESCRIPTION)
A table appears with rows for Image, 1, 2, 3, three dots, and percentage of total.

(SPEECH)
So if carrying out error analysis to evaluate these three ideas, what I would do is create a table like this.

And I usually do this in a spreadsheet, but using an ordinary text file will also be okay.

And on the left side, this goes through the set of images you plan to look at manually.

So this maybe goes from 1 to 100, if you look at 100 pictures.

And the columns of this table, of the spreadsheet, will correspond to the ideas you're evaluating.

So the dog problem, the problem of great cats, and blurry images.

And I usually also leave space in the spreadsheet to write comments.

So

(DESCRIPTION)
The columns are hand-written in the table, the labels are, dog, great cats, blurry, and comments. dddddd

(SPEECH)
remember, during error analysis, you're just looking at dev set examples that your algorithm has misrecognized.

So if you find that the first misrecognized image is a picture of a dog, then I'd put a check mark there.

(DESCRIPTION)
A checkmark is placed on the cross of the numeral 1, and the column, dog. Subsequent checkmarks are made at the corresponding points on the table.

(SPEECH)
And to help myself remember these images, sometimes I'll make a note in the comments.

So maybe that was a pit bull picture.

If the second picture was blurry, then make a note there.

If the third one was a lion, on a rainy day, in the zoo that was misrecognized.

Then that's a great cat, and the blurry data.

Make a note in the comment section, rainy day at zoo, and it was the rain that made it blurry, and so on.

Then finally, having gone through some set of images, I would count up what percentage of these algorithms.

Or what percentage of each of these error categories were attributed to the dog, or great cat, blurry categories.

So maybe 8% of these images you examine turn out be dogs, and maybe 43% great cats, and 61% were

(DESCRIPTION)
The three percentages are marked at the bottom of the table.

(SPEECH)
blurry.

So this just means going down each column, and counting up what percentage of images have a check mark in that column.

As you're part way through this process, sometimes you notice other categories of mistakes.

So, for example, you might find that Instagram style filter, those fancy image filters, are also messing up your classifier.

In that case, it's actually okay, part way through the process, to add another column like that.

For the multi-colored filters, the Instagram filters, and the Snapchat filters.

And then go through and count up those as well, and figure out what percentage comes from that new error category.

The conclusion of this process gives you an estimate of how worthwhile it might be to work on each of these different categories of errors.

For example, clearly in this example, a lot of the mistakes we made on blurry images, and quite a lot on were made on great cat images.

And so the outcome of this analysis is not that you must work on blurry images.

This doesn't give you a rigid mathematical formula that tells you what to do, but it gives you a sense of the best options to pursue.

It also tells you, for example, that no matter how much better you do on dog images, or on Instagram images.

You at most improve performance by maybe 8%, or 12%, in these examples.

Whereas you can to better on great cat images, or blurry images, the potential improvement.

Now there's a ceiling in terms of how much you could improve performance, is much higher.

(DESCRIPTION)
The percentages 43 percent, and 61 percent, are circled.

(SPEECH)
So depending on how many ideas you have for improving performance on great cats, on blurry images.

Maybe you could pick one of the two, or if you have enough personnel on your team, maybe you can have two different teams.

Have one work on improving errors on great cats, and a different team work on improving errors on blurry images.

But this quick counting procedure, which you can often do in, at most, small numbers of hours.

Can really help you make much better prioritization decisions, and understand how promising different approaches are to work on.

So to summarize, to carry out error analysis, you should find a set of mislabeled examples, either in your dev set, or in your development set.

And look at the mislabeled examples for false positives and false negatives.

And just count up the number of errors that fall into various different categories.

During this process, you might be inspired to generate new categories of errors, like we saw.

If you're looking through the examples and you say gee, there are a lot of Instagram filters, or Snapchat filters, they're also messing up my classifier.

You can create new categories during that process.

But by counting up the fraction of examples that are mislabeled in different ways, often this will help you prioritize.

Or give you inspiration for new directions to go in.

Now as you're doing error analysis, sometimes you notice that some of your examples in your dev sets are mislabeled.

So what do you do about that?

Let's discuss that in the next video.