Size of the dev and test sets

(SPEECH)
>>

(DESCRIPTION)
Logo, Deep Learning.ai. Setting up your goal. Size of dev and test sets

(SPEECH)
In the last video, you saw how your dev and test sets should come from the same distribution, but how long should they be?

The guidelines to help set up your dev and test sets are changing in the Deep Learning era.

Let's take a look at some best practices.

You

(DESCRIPTION)
Old way of splitting data. Draws long rectangle and divides into sections.

(SPEECH)
might have heard of the rule of thumb in machine learning of taking all the data you have and using a 70/30 split into a train and test set, or have you had to set up train dev and test sets maybe, you would use a 60% training and 20% dev and 20% tests.

In earlier eras of machine learning, this was pretty reasonable, especially back when data set sizes were just smaller.

So if you had a hundred examples in total, these 70/30 or 60/20/20 rule of thumb would be pretty reasonable.

If you had thousand examples, maybe if you had ten thousand examples, these things are not unreasonable.

But in the modern machine learning era, we are now used to working with much larger data set sizes.

So let's say you have a million training examples, it might be quite reasonable to set up your data so that you have 98% in the training set, 1% dev, and 1% test.

And when you use DNT to abbreviate dev and test sets.

Because if you have a million examples, then 1% of that, is 10,000 examples, and that might be plenty enough for a dev set or for a test set.

So, in the modern Deep Learning era where sometimes we have much larger data sets, It's quite reasonable to use a much smaller than 20 or 30% of your data for a dev set or a test set.

And because Deep Learning algorithms have such a huge hunger for data, I'm seeing that, the problems we have large data sets that have much larger fraction of it goes into the training set.

(DESCRIPTION)
Size of test set. Set your test set to be big enough to give high confidence in the overall performance of your system.

(SPEECH)
So, how about the test set?

Remember the purpose of your test set is that, after you finish developing a system, the test set helps evaluate how good your final system is.

The guideline is, to set your test set to big enough to give high confidence in the overall performance of your system.

So, unless you need to have a very accurate measure of how well your final system is performing, maybe you don't need millions and millions of examples in a test set, and maybe for your application if you think that having 10,000

examples gives you enough confidence to find the performance on maybe 100,000 or whatever it is, that might be enough.

And this could be much less than, say 30% of the overall data set, depend on how much data you have.

For some applications, maybe you don't need a high confidence in the overall performance of your final system.

Maybe all you need is a train and dev set, And I think, not having a test set might be okay.

(DESCRIPTION)
Draws long rectangle, splits train and test

(SPEECH)
In fact, what sometimes happened was, people were talking about using train test splits but what they were actually doing was iterating on the test set.

So

(DESCRIPTION)
Crosses out test, writes dev

(SPEECH)
rather than test set, what they had was a train dev split and no test set.

If you're actually tuning to this set, to this dev set and this test set, It's better to call the dev set.

Although I think in the history of machine learning, not everyone has been completely clean and completely records of about calling the dev set when it really should be treated as test set.

But, if all you care about is having some data that you train on, and having some data to tune to, and you're just going to shake the final system and not worry too much about how it was actually doing, I think it will be healthy and just call the train dev set and acknowledge that you have no test set.

This a bit unusual?

I'm definitely not recommending not having a test set when building a system.

I do find it reassuring to have a separate test set you can use to get an unbiased estimate of how I was doing before you shift it, but if you have a very large dev set so that you think you won't overfit the dev set too badly.

Maybe it's not totally unreasonable to just have a train dev set, although it's not what I usually recommend.

So to summarize, in the era of big data, I think the old rule of thumb of a 70/30 is that, that no longer applies.

And the trend has been to use more data for training and less for dev and test, especially when you have a very large data sets.

And the rule of thumb is really to try to set the dev set to big enough for its purpose, which helps you evaluate different ideas and pick this up from AOP better.

And the purpose of test set is to help you evaluate your final cost buys.

You just have to set your test set big enough for that purpose, and that could be much less than 30% of the data.

So, I hope that gives some guidance or some suggestions on how to set up your dev and test sets in the Deep Learning era.

Next, it turns out that sometimes, part way through a machine learning problem, you might want to change your evaluation metric, or change your dev and test sets.

Let's talk about it when you might want to do that.