WEBVTT

1

00:00:04.640 --> 00:00:13.470

Jisun An: Alright welcome back to the lecture. Today's pass code is Dpo and So

2

00:00:13.550 --> 00:00:34.590

Jisun An: I've updated the the team rosters for practical assignments and also the projects. I mean, mostly you don't need to mostly worry about it. There were some recast, so I changed it. A few teams, but just double check. If you, your teammates may have been, change it. So just make sure. I I

3

00:00:34.610 --> 00:00:54.479

Jisun An: I think there will be no further changes, but but just in case. So check again and for the practical assignment. Please start only to communicate with your teammates to create a data set and do the the the next steps.

4

00:00:55.740 --> 00:01:05.799

Jisun An: And so in 2 weeks we will have the like proposal presentation for the project. I will

5

00:01:06.600 --> 00:01:30.981

Jisun An: create the the rubrics, and then we'll share hopefully by the end of like, only next week, so that you have a better guidance to. But I will. I mean, the scope will be quite broad. So I think any projects that you are currently thinking should be okay. But the rubric just consider it as a guidance for you to help out to reshape your presentation.

6

00:01:31.660 --> 00:01:33.750

Jisun An: so that would be something.

7

00:01:36.380 --> 00:01:39.000

Jisun An: Yep, I think that's all, for now.

8

00:01:41.270 --> 00:01:47.220

Jisun An: So let me move back to the where we left from the

9

00:01:48.710 --> 00:02:15.640

Jisun An: reinforcement learning from human feedback. So I just want to recap this part because I'm sorry I was using this figure, and I suddenly kind of block out a little bit what this actually meant for but I just want to recap so so that you have still some understanding about how these policies are updating, and I will talk a little bit more later. but so so in this particular figure, these.

10

00:02:15.640 --> 00:02:45.000

Jisun An: they had this neural network for policy. So the input here is the state, and the output is the course of the actions or possible actions. And basically, this network will give you given a state, what is the probability that that your next actions? So that's the what would be like the input and the output of this eventual policy neural network. And this slide wanted to explain how this network can be updated

11

00:02:45.000 --> 00:03:09.769

Jisun An: and the the way that it updates is, firstly, it's take a 1 random path. So start from the state 0 and it goes to the end of the state where they will get the actual rewards. So this was the one of such examples. So it started from this one cell, and then and then it just like go through following the policy network. So

12

00:03:10.180 --> 00:03:16.959

Jisun An: how can they find this path? I mean the whatever the current policy network has. So even though the

13

00:03:17.120 --> 00:03:30.260

Jisun An: the action probability. For even for those given States this may not be optimal, but whatever you have at the moment with that policy network, you can follow those paths. So at when he started from this position.

14

00:03:30.300 --> 00:03:57.660

Jisun An: given the current policy network, it says, Go to the left. And then, when they were in the state of 2. 1. It says, the most probable action must go up, so they were kind of like following that actions, and then they reach to the state where they get they can get the rewards. Then, once you can get reach to the state where you get the rewards, then you can recompute what are the expected rewards that you can get for each of those

15

00:03:57.660 --> 00:04:06.189

Jisun An: States on the path? Because you now already know the path, and you know the eventual reward. So you can redistribute or recompute these

16

00:04:06.460 --> 00:04:31.330

Jisun An: values. So then you can kind of create for each of these paths you are creating this data points for training your neural network. So, for example, if we are just looking at this one particular example, it says that when the state is the 4 5, when the when the agent is here, the most probable action was the right and the policy network. So this

17

00:04:31.330 --> 00:04:56.229

Jisun An: was the current policy, meaning that the probability of taking that particular action. So the currency current policy says, that is, the probability is like the 0 point 3, and then in the by taking this action, these resulting in, get the reward and the gain was very high, which is 4. So in this case we know that, taking the action.

18

00:04:56.390 --> 00:05:18.139

Jisun An: meaning that going to the right should be encouraged, boosted within this neural network, right? So the all the parameters in the network will be tuned to boost this particular action, and that means the I mean the low part you can ignore for now, so the change they are encouraging to increase this particular probability of the action.

19

00:05:18.420 --> 00:05:22.720

Jisun An: and in the other end, if you're looking at this other example

20

00:05:23.108 --> 00:05:42.851

Jisun An: for this particular state 3, 1 eventually they took the action which was the left I mean going to the left and the prop, because they the probability, was fairly high, like 0 point 7 but then they found out eventually, if they take left from here, then their eventual reward,

21

00:05:43.210 --> 00:06:04.230

Jisun An: computed expected value is like minus 3. So once again, this we had a rule that even though you get the reward, every one step, it costed one. So basically, taking long path may give you some penalty. So this, this minus 3 working as a penalty. So this indicates now that okay, taking left is actually not the best idea. So let's

22

00:06:04.400 --> 00:06:17.066

Jisun An: take not that action. So that's the signaler that this data can provide to the network. So that means that, okay, so it need to be, this probability need to be decreased. So this is a sign that

23

00:06:17.620 --> 00:06:27.889

Jisun An: this particular example can get and give to the neural networks. And then, if you are seeing now all of them, then, for each of these States action pairs.

24

00:06:27.890 --> 00:06:52.530

Jisun An: we are computing a particular gain and using that gain. We are adjusting the neural networks, parameters to whether to boost that particular action or deboost that particular action. So that's the what, how the policy neural network would updates, I mean, even though this is like very, very simplified version. But it's the how the Rl. Within Rl, how the policy neural network would be

25

00:06:52.530 --> 00:07:21.190

Jisun An: updated. And we also talked about the value neural networks. And there's value. Neural networks is relatively easy, because, I mean, you can literally compare with the actual value. So they just compute, compare the current value and the actual value, and then they just using the subtraction of the value as of like loss is, it is. But then the policy network might be slightly more tricky. But I think this is some quite straightforward way to think how the this policy networks will be updated.

26

00:07:22.190 --> 00:07:31.829

Jisun An: So I will talk a little bit little more about this Rl. Later of the next lecture, but I just wanted to remind and also clarify what I meant for for this.

27

00:07:31.950 --> 00:07:35.820

Jisun An: Any any questions from on this slide in particular.

28

00:07:40.580 --> 00:07:57.939

Jisun An: Okay? So so we were talking about like high, level overview of why Llrl is needed for the language modeling. And we discussed about why, rl, how rl has been used very briefly.

29

00:07:58.150 --> 00:08:23.440

Jisun An: and now we are talking a little bit more about the Rlhf. And it. As I mentioned, they usually have 3 steps. So the 1st step is so they need to. They start from the instruction tuned models. And then they in the second step. They are building their reward model from the preference data set. And thirdly, they actually run the Rl based on this reverse model. So these are the 3 steps, and I will go through that today.

30

00:08:23.450 --> 00:08:51.179

Jisun An: So I mean this already we talked about. So I will just move on. So so the preference rating is something that coming from the human. So we are asking. So here, something that you need to be really clear is we are sampling different outputs. Given the same prompt. So because Lim is just a probabilistic models, even for the same prompt. The output may be slightly different time to time.

31

00:08:51.180 --> 00:09:11.179

Jisun An: If you are not using the greedy sampling. I mean, if you're using greedy decoding that the answer will be always the same. But if it's not, if you are doing sampling, then you will get like a different sample, different outputs. So the preference ratings here, so given, the same, prompt to the given that we have instruction tuned at them. We are sampling like different outputs.

32

00:09:11.180 --> 00:09:22.480

Jisun An: given the same prompts, and for each of these outputs we ask human to rate them so which output you prefer, and our goal is to

33

00:09:22.630 --> 00:09:28.739

Jisun An: tune our edit M to adjust. Based on this preference, adapt to this preference.

34

00:09:29.530 --> 00:09:56.440

Jisun An: So so there could be like a different ways. But one of the 1st attempts was giving multiple outputs, and then they asked the human to rank them like from one to 5, and then they used it. They called it as a preference judgment, and they used it for their training. But here one of the limitations of collecting this human feedback is basically you need to hire someone to do it, and it would be very costly, right? And especially

35

00:09:56.550 --> 00:10:24.480

Jisun An: to train the model with the rl, it will require a lot of steps and a lot of trainings, so meaning that even for a very small number of prompts you will need a lot of human ratings. So the idea here is because basically it's in part because because you can generate lots of outputs from the evidence, right? Even for the same prompt. You can generate like 1 1,000 different outputs, and then you can. I mean, you can ask like preference to human.

36

00:10:24.620 --> 00:10:37.950

Jisun An: But then, just just to do that labeling to be very costly. So the idea is rather than asking human. Let's just build a model that that can mimic the preference of the human. And that's the what we call this a reward model.

37

00:10:38.670 --> 00:11:06.069

Jisun An: So the reward model is, so can we train the model to predict this human preference, judgment. And the input is the prompts. And they're one of the outputs. So that's the input and the output is the scalar. Some scalar score that represents the preference. So preference score that, given this prompt and the output pair, what is the preference score? So that's the what the reward model will be trained using the preference data set

38

00:11:07.210 --> 00:11:23.950

Jisun An: and so there could be many different way to model the preference. But the Rhf paper initially used this particular model which is called as a Bradley Terry hairwide preference model.

39

00:11:24.080 --> 00:11:32.900

Jisun An: The idea is quite simple. So, assuming that we have the X like the usual prompts, and we now have. We assume there are 2 outputs.

40

00:11:32.940 --> 00:11:52.320

Jisun An: And then we already asked the human, which one is you like, which one you prefer, the more so we assume that Yw. Is the the one that preferred, and we then we also have this Yl. Which is less preferred response. So once we ask users to rank all these 5 like outputs.

41

00:11:52.320 --> 00:12:12.100

Jisun An: Then you can pick just 2 output, and you can tell which one is more preferred or not right? So like pairwise, preference is something easy and easy to train, so I think this was the 1st attempt that they were using. But there are also some other ranking models, and they also have been trying some other models as well, and it works as similar as this model was.

42

00:12:13.000 --> 00:12:37.586

Jisun An: And then so these are some of the variables that we are defining, and then R is the our reward function that are assigning. This is color score, then the probability that response Yw. Is preferred over Dyl in terms of their reward score can be defined in this way. So I mean, this is the preference model

43

00:12:38.280 --> 00:12:47.529

Jisun An: and even though this equation, I mean, anyone feels that this equation looks familiar or seem I mean familiar looks familiar to you.

44

00:12:49.190 --> 00:13:04.110

Jisun An: So these are e to the reward of given X and Yw. Divided by exponentiated reward XW. Yw plus exponent reward X given X and YL.

45

00:13:04.230 --> 00:13:07.079

Jisun An: Is this equation looks familiar to you.

46

00:13:11.060 --> 00:13:15.950

Jisun An: I think if you see it, it'll report quite clear. But this is basically very similar to the soft nets.

47

00:13:16.490 --> 00:13:26.700

Jisun An: So what it does is that so basically, the R function will give you the reward score. But then, so we just want to turn that score into the probability.

48

00:13:26.700 --> 00:13:54.100

Jisun An: So basically, you can simply think this. As so, we have the 2 rewards scores. And then we are basically normalizing the reward score over the I mean, basically the preference score for the Yw. Divided by the sum of these 2. But then, because these are the just, the numbers, we would just want to turn them into the probability. And we are just taking exponent exponential here. So this. And this is basically why, what the softmax actually does.

49

00:13:55.568 --> 00:14:02.580

Jisun An: Very similar to that. So so this is the Bradley Terry, pairwise preference model.

50

00:14:03.580 --> 00:14:18.770

Jisun An: And now, once we have a probability model, then we can just turn the I mean we can simply take the get the loss from it by taking the low, negative, low likelihood method. Oh, before that. So

51

00:14:19.350 --> 00:14:43.969

Jisun An: so this simply tells that if the preferred response reward score is greater than the reward score of the less preferred response, then the probability approaches one. If they are similar, then it's close to 0 point 5, and if it's the other way around, then probability approaches 0. So it encourages to that. The reward score of the preferred response. Have the higher score.

52

00:14:44.816 --> 00:14:46.560

Jisun An: By by this model.

53

00:14:46.920 --> 00:15:02.919

Jisun An: and once you have this probability model, then you can also get the loss function, which is a simple, negative, low likelihood loss which can be defined now as a sigmoid form, and subtracting the one reward from the other.

54

00:15:02.920 --> 00:15:18.220

Jisun An: and the actual derivation is in the next page, and I think this is fairly, relatively simple derivation, so you can follow it. But I will describe it briefly and take a look at it later. So we start from this

55

00:15:18.220 --> 00:15:22.050

Jisun An: Bradley preference model, and

56

00:15:22.240 --> 00:15:39.729

Jisun An: you are factor out this ER. The reward one of this term from the denominator and factor out means that you are basically dividing these terms by this particular term, ER, x and yl.

57

00:15:39.730 --> 00:15:58.384

Jisun An: so so if you are dividing this entire equation with this term, then it will rearrange it in this way. And now this equation looks very similar to the sigmoid function where the Sig mode is one divided by one plus e to d minus jet

58

00:15:59.070 --> 00:16:22.739

Jisun An: so you can. Now, basically re, like, basically, this itself is the can be re presented as a sigmoid function. So now the jet became so. If we replace the this jet as this the subtraction of these 2 reverse, then these are like basically like the sigmoid function.

59

00:16:23.780 --> 00:16:51.539

Jisun An: So now that we know that this probability also can represent it as a sigmoid function, or that the results are basically the following, the sigmoid distribution, then we are simply taking the we want to maximize the likelihood of this wy over. Wl, then we take the negative low likelihood to get the loss function, and that will be your final loss. Function, that where you are seeing in a previous slide

60

00:16:52.398 --> 00:17:21.330

Jisun An: so once again, this is I think I mean at at a glance it look. It may look a bit tricky, but take your time, and I think it's it's relatively easy to understand. So so overall. So this will be our loss function. So once we once we because we have a probability to define when we have a preference data set. And also we can define a loss function. So now you can simply train a model using this loss function right?

61

00:17:25.839 --> 00:17:29.340

Jisun An: Any custom here like.

62

00:17:42.010 --> 00:17:50.120

Jisun An: oh, because we are taking. Oh, because the sigmoid is the one over one plus EE minus jet.

63

00:17:50.580 --> 00:17:58.090

Jisun An: Oh, so oh, oh, oh, the it should be the other way around. Yeah, yeah, you're right. So. Oh, oh, you're right

64

00:17:59.520 --> 00:18:00.580

Jisun An: then.

65

00:18:06.901 --> 00:18:20.380

Jisun An: No. Actually, this is correct form, because, these are XYW minus rx comma yl, it itself is deject. So it just we are adding.

66

00:18:21.250 --> 00:18:23.065

Jisun An: so we are not.

67

00:18:30.030 --> 00:18:32.070

Jisun An: okay, thanks. You guys.

68

00:18:55.660 --> 00:19:00.559

Jisun An: yeah, because you had this term here. So you yeah, you need to divide it as well, sir.

69

00:19:01.795 --> 00:19:27.830

Jisun An: yeah. So once we have this loss, then basically, we can train our reward model. But once again, I mean, I will not gonna ask you to drive this thing. But the key here is that we can. By following this loss function. Given that, we have a preference data set, we can build this reward model, where basically, the input is the any, any problems and the output. And then the output will be the scholar score.

70

00:19:29.390 --> 00:19:40.272

Jisun An: so we want to. The reward model will be like given Xn a particular output that is preferred. If we are going through this

71

00:19:41.110 --> 00:20:00.969

Jisun An: instruction tuned model, then we will have like define our outputs. Then we basically want to train this model to get the single scholar function right. And the way that we can do. There could be various way that you can do. But one way is basically you are, you may just take this last vector of these encoded text.

72

00:20:00.970 --> 00:20:15.919

Jisun An: And then you just add one extra layer, where, by combining the last factor and these weight metrics, then it basically turns into a singular single scholar. So eventually this will be a reward function.

73

00:20:16.250 --> 00:20:27.559

Jisun An: and even though I don't have it in this figure, but when you are actually training, the loss itself is coming from the subtracting from the preferred response a reward of the preference

74

00:20:27.560 --> 00:20:46.510

Jisun An: preferred response minus reward of the less preferred response. So you will use that value to backdrop, and even once this model is trained, then you may get given this X and yw, you will get like 8.0, which is this color score, so that will be the what this reward function will give it to you.

75

00:20:47.050 --> 00:20:48.690

Jisun An: Does that make sense?

76

00:20:49.660 --> 00:21:07.320

Jisun An: So after like long good? Especially, I think you will need a good data set for this good preference data, and once you have it, this reward model will predict the human preference. And now this will use to provide the rewards to the Rl models.

77

00:21:10.390 --> 00:21:11.180

Jisun An: Okay.

78

00:21:16.490 --> 00:21:35.308

Jisun An: So now that these are now the trend model, then for the same prompt, we can sample different outputs and any any outputs we can basically get some going through to the reward model, and then we can get the their reward values. Then this will now get back to our

79

00:21:37.070 --> 00:22:06.080

Jisun An: rm, model. So then, and so now, once we have this reward model, there could be several different way that how you can use these human preference to be to align our evidence. And so and maybe rl, may not be. Rl, basically is not the only option, but there could be some other options. So one would be best of and sampling, and it's also called as a rejection sampling, and

80

00:22:06.380 --> 00:22:30.140

Jisun An: what it does is given a prompt, you are, generate N samples. And then, basically, you are using this reward model for each of the outputs. You can now get a score from this reward model. And then, basically, you are choosing the sample with the highest reward. So, in other words, whenever you are in the inferring stage of the Rrm. You are just doing this so for

81

00:22:30.140 --> 00:22:55.069

Jisun An: every and any prompts that you are giving, you are generating multiple output, evaluating, using the reward model, and then just selecting the best one, and then replying back to the users. So obviously, this is the I mean, this may work right. But then the problem is, it's computationally very expensive, because for every prompt you will need to do n, some. I mean, n, inference. So the inference cost will basically n times higher.

82

00:22:59.660 --> 00:23:09.969

Jisun An: And then then maybe the second question is, why then not? Why don't we just fine tune our model with this preference data. I mean, that's also one option. Right?

83

00:23:10.290 --> 00:23:29.670

Jisun An: So we can generate a large data set, meaning that for various prompts we can generate various outputs where each of the prompt and the output pair can have the reward value. Using our reward model, then we can use it as the fine tune data, so we can generate this data and fine tune, our model.

84

00:23:30.240 --> 00:23:58.080

Jisun An: That's also possible. But basically, firstly, it will probably requires a really large data. But if we are just using the models then creating large data set may not be that difficult in this case, because it doesn't require any human environment here, but the fine tuning, one of the fine tuning, neglecting the aspects of learning from negative samples. So

85

00:23:58.290 --> 00:24:18.389

Jisun An: basically, if we are basically if we are tuning the model. Using this this kind of steps, then we will all the parameters will be tuned to maximize for the like more preferred responses, and they will ignore the less preferred responses. But then,

86

00:24:19.090 --> 00:24:47.806

Jisun An: so basically. But then there may be something to learn from these negative samples, because these are eventually all the output that coming from the model. So we need to learn what to and as we talked about from the policy neural networks like. We want to learn, like what what action should be boosted when action should be like discouraged. So, but basically no more fine tuning is not very easy to do that, so that would be the one kind of

87

00:24:48.430 --> 00:24:59.980

Jisun An: points that are missing out. So instead of these 2 approaches, now, we are using this reinforcement learning. So for increase, this probability of

88

00:25:00.110 --> 00:25:17.480

Jisun An: getting the more preferred outputs. Given the X slightly more and decrease this probability to get the less preferred output. Given the slightly where these amounts, how much we want to increase or decrease our function of the rewards themselves.

89

00:25:17.480 --> 00:25:46.630

Jisun An: So once again, think back again the policy network updating the policy network that we talked about. So once we know the reward, we will know whether that action need to be encouraged or discouraged. But then we also want to know how much we want to encourage and discourage it, and that amount is coming from the reward. So the gain that you've seen in the previous figure here. That gain is the estimated value or estimate estimated rewards.

90

00:25:48.350 --> 00:26:00.600

Jisun An: So that's the why we are going with the reinforce enforcement learning. So the and

91

00:26:03.370 --> 00:26:09.890

Jisun An: So the one thing that also to note is that, or making the rl, is

92

00:26:13.690 --> 00:26:14.410

Jisun An: whether

93

00:26:15.510 --> 00:26:34.209

Jisun An: one thing to notice that, so we only observe these rewards only after generating a complete sequence. So the preferences on the sentences right? So. But then, in the rl, we are kind of creating, generating one token by one token, so you will take some time, but we only observe the reward at the end of the sequence.

94

00:26:34.600 --> 00:26:47.110

Jisun An: and but then the Rl. Will propagate the final reward to all the intermediary steps and guiding the model towards the optimal token sequences, is something that we've seen from the below figure.

95

00:26:47.420 --> 00:26:59.730

Jisun An: and the policy here refers to the probability distribution of the tokens, sequences that the model generates so so

96

00:27:00.900 --> 00:27:23.470

Jisun An: so to go to. What is the how, how the Ra will be trained let me. So we have like 2 different policies. So one is the we have this current policy the reference policy that we are using. And then we also have the new policy that the current policy that we will learn and updates.

97

00:27:23.470 --> 00:27:40.691

Jisun An: So here the reference policy is the instruction tuned model that we already have. And so we will have it as a reference model, because the reason is, rl, learning sometimes can be very

98

00:27:41.270 --> 00:28:00.549

Jisun An: like sporadic and spontaneous. And it could be there could be have like big jumps. So basically, the training can be very unstable. So, and and they found that especially without controlling by learning through the Rl, the language model just became changing like too fast.

99

00:28:00.550 --> 00:28:27.449

Jisun An: So they want to regularize, added some kind of regularization term, and for doing so they want to have some reference model. So which is our instruction tuned. So this instruction tuned model is already quite good model, right? It's trained with a large data set and also trained to follow the instructions. So we have it as a reference model. And when we are updating our L model, which is the the pi, the current policy model.

100

00:28:27.900 --> 00:28:35.580

Jisun An: we are kind of trying to not to deviate too much from the reference model one second.

101

00:28:36.490 --> 00:28:37.449

Jisun An: isn't that neat?

102

00:28:39.370 --> 00:29:04.609

Jisun An: Yes, for sure. So basically, if we have, like Lama 7 B model. Then we will at least have 2 models here to have the reference model and the updated model itself, and also the gradient as well. So so it takes a lot of time memories. And so our is very, very expensive, expensive to train.

103

00:29:08.650 --> 00:29:13.949

Jisun An: So and so that's the where these 2 terms are coming from.

104

00:29:14.400 --> 00:29:21.430

Jisun An: And then this is the actual, the optimization objective and I mean, so

105

00:29:21.820 --> 00:29:39.740

Jisun An: firstly, these, this is not the loss. This is the objective. So we want to maximize this term. So what we want to do is our. So the goal is to find the best policy. So the policy is the probability distribution given a State. In other words, that's the like. What determines the next tokens right?

106

00:29:39.740 --> 00:30:09.410

Jisun An: So we want to find the optimal policy that maximize this particular value. And do I have so here the 1st term that we are maximizing is the rewards I mean. Obviously, this is, we are doing all to maximize the incorporated human preference. And this human preference is coming from this rewards. So we want to maximize this rewards model. And at the same time we want to prevent the model deviating too much from the reference model. So we are having this another term

107

00:30:09.853 --> 00:30:28.650

Jisun An: that are having the negative value. So basically, we want to minimize to maximize this objective. So we want to maximize the reward. But at the same time, we want to minimize this divergence. Kl, divergence to be, not to be to change it from the reference model.

108

00:30:28.910 --> 00:30:51.920

Jisun An: And here this Kl divisions divergence term is defined as this. So it's a simply, we are basically comparing the outputs. I mean the next token given the current outputs divided by the reference models. So we are simply computing, I mean, comparing the 2 distribution of the reference models and the current policy models.

109

00:30:53.110 --> 00:31:13.550

Jisun An: And then so and so this is, I mean, it's as simple as it is. These are the optimization objective. And once you have this, you can use either like any method for training the Rl, like proximer policy optimization algorithm, which is called as a Ppo or the reinforce, which is very niche kind of Rl, algorithms.

110

00:31:13.900 --> 00:31:41.899

Jisun An: And the Ppo is something. So we already talked about policy networks and the value networks. The idea here, I mean, and there are. There are different methods that are like just just learning the policy networks or the value networks. But the Ppo is idea is basically they want to train the 2 networks at the same time. So they just updating the both networks same and that that's the what Ppo kind of does. But once again, I mean

111

00:31:42.000 --> 00:31:52.139

Jisun An: to do that it will require like few more lectures. And this is a bit of out of scope. So we're not going to go details about it. But so once, basically, that will be something.

112

00:31:54.000 --> 00:31:56.819

Jisun An: That we'll see. So

113

00:31:56.950 --> 00:32:19.799

Jisun An: thinking back to that policy network updates. So we only that example had a very simple reward. Right? It has a reward term, and then they only have a cost for the steps. But then, in our case this objective will be the something optimized for. So this will basically determine the the rewards, and how much expected values should be should be there.

114

00:32:26.420 --> 00:32:28.359

Jisun An: So any customs

115

00:32:31.485 --> 00:32:34.959

Jisun An: is the supervised fine tuned.

116

00:32:35.580 --> 00:32:48.600

Jisun An: So this is the instruction model instruction tuned model. So I will interchangeably using these 2. So it's the supervised fine tuned model which is the instruction instruction tuned model. It's the same same words. Yeah.

117

00:32:51.250 --> 00:32:54.364

Jisun An: So the only thing that you should remember is

118

00:32:55.620 --> 00:33:24.029

Jisun An: Once again, I will. I will do, have a little bit more on the Rl. Later, but especially in the Rl. Etf, we want to maximize the rewards which is the basically reflecting the human preference. So whenever we are selecting the next token, we want to choose the token that increase the human preference. But then, at the same time, we don't want to deviate too much from the original model, our instruction tuned.

119

00:33:24.070 --> 00:33:38.200

Jisun An: So we have that divergence term which compares the 2 distributions and and using it to penalize to the values, so that so that the updates will not be very dramatic.

120

00:33:44.950 --> 00:34:00.730

Jisun An: So in just some overview of this Rlhf pipeline. So we have this space pre-trained. Lm, so these are tuned based on the unsupervised way without any labeled data. It was only using just the text.

121

00:34:00.730 --> 00:34:15.390

Jisun An: And then and then we instruction tuned, using the instruction data set, which is pairs of question and the answers. And these answers were written by the experts or the human labelers, so that we tune our model to follow

122

00:34:15.699 --> 00:34:36.099

Jisun An: what we want, based on some of the instructions and some of our preference there as well. So we had together this data set of instruction and the input and output and various inputs and the outputs. And we tune them. And then using this instruction tuned model, we fine tune again to

123

00:34:36.230 --> 00:35:01.900

Jisun An: build our reward model, using a new data set which we called as a preference data set where it has given an X given and 2 outputs. We basically had a label which of these outputs were preferred and then based on that preference data set, we trained our reward model. And now we use these 2, our instruction tuned model and the reward model. We had the another

124

00:35:01.900 --> 00:35:18.970

Jisun An: model which call that we called as a policy model. That's the our eventual. So the policy model is exactly the same as the instruction tuned model in terms of the architecture. It will be like a copy of the instruction tuned model. But we will just update that parameter based on this new objective that we had in the Rl.

125

00:35:19.140 --> 00:35:25.600

Jisun An: So we will train the Ppo with the Ppo to maximize this expected rewards that we had

126

00:35:26.230 --> 00:35:37.094

Jisun An: and then that would be our online model. And this was the overall pipeline that were proposed by the Gpt open AI when they were

127

00:35:38.550 --> 00:35:52.152

Jisun An: We're leaving the Chatgpt. 3.5, I think. So that's which version that you've seen. So that was the after going through these 3 steps. Pre-training, instruction, tuning, and then preference tuning on preference.

128

00:35:53.100 --> 00:35:56.550

Jisun An: preference reinforcement learning, I mean. Rl, hf.

129

00:35:58.120 --> 00:36:19.715

Jisun An: so so here. So I I talk a little bit about the Rl. Itself, but I don't want you to understand everything, because it was very brief introduction to it. But at least you should understand the flow of this, and especially how the reward model was tuned, and also how that has been used for the Rl. H.

130

00:36:20.430 --> 00:36:25.690

Jisun An: model. And what were the kind of the data set that used any any questions?

131

00:36:27.240 --> 00:36:37.990

Jisun An: Ppo. Ppo is, I also need to look back. Proximal policy optimization algorithm here, yeah.

132

00:36:45.810 --> 00:36:46.820

Jisun An: any question.

133

00:36:52.170 --> 00:36:52.900

Jisun An: Okay?

134

00:36:53.340 --> 00:37:01.270

Jisun An: So I think this was the last slide for the Rlatf and I will move on to the next one. So

135

00:37:01.470 --> 00:37:12.110

Jisun An: the Rlhs has been used. But still, as I told you, it has been very black box. The the detailed settings are not

136

00:37:12.230 --> 00:37:20.370

Jisun An: very well exposed, but and also our usually takes a long time, and it has a lot of noise as well. So it would be

137

00:37:20.890 --> 00:37:25.470

Jisun An: like the training, the exact. The same thing may not be very easy

138

00:37:26.950 --> 00:37:29.849

Jisun An: and so, followed by this

139

00:37:31.390 --> 00:37:53.749

Jisun An: Rlhtf, there has been a few interesting development to incorporate the human preference but without rl or with rl, but is in the easier kind of way. So I will talk about these 2 if time permits. But let me start with the Dpo, the direct preference optimization.

140

00:37:54.330 --> 00:38:14.639

Jisun An: So let me go back to this Rlhf optimization objective. And once again this one was for the maximizing, the reverse. This one was preventing the model from deviating from the reference model, and the pi is the current aligned rlm, and this pi is keep updating over the Rl. Process.

141

00:38:15.120 --> 00:38:24.360

Jisun An: The Pi reference is the our instruction tune, the model or the reference model that we are keeping it. So we are not changing this reference because that's just a reference model.

142

00:38:24.550 --> 00:38:31.679

Jisun An: So the idea of the Dpo or the problem of this Rlhtf is, there are 2 things. One is.

143

00:38:32.510 --> 00:38:35.350

Jisun An: can we get rid of this reward? So

144

00:38:35.570 --> 00:38:58.760

Jisun An: training the reward model, I mean, basically predicting human preference is not easy. I mean. Still, it is depending on the fine tuning which requires a preference data. So having a good reward model means that you need a good reward preference data set. But then and it may. And even though you have a larger data set, would it be ever possible to tell that it will really predict the human preference.

145

00:38:58.760 --> 00:39:06.050

Jisun An: So having the reward model is making everything unstable, and at the same time is also very expensive as well.

146

00:39:06.490 --> 00:39:18.649

Jisun An: So the idea is like, so can we get rid of these rewards? And I mean not we are not. We get rid of it. But can we just turn them into a probability and not using it in as a part of this rl, or

147

00:39:19.170 --> 00:39:47.220

Jisun An: Rl itself. And the second idea was that can we also avoid the rl, entirely, and the reason is that the rl, it has this rollout rollout is basically you need to for each of the step you need to infer the next token. And then you are computing all this, and then you are spending expending the another next token. So you you need to

148

00:39:48.307 --> 00:39:50.219

Jisun An: expand or have

149

00:39:50.660 --> 00:39:58.220

Jisun An: have state and the action pairs until you reach to the reward. And you you need that for every steps of the

150

00:39:59.323 --> 00:40:10.239

Jisun An: training. And that's the actually quite expensive expense, I mean, some inferencing stuff is already expense. And to train something based on the inference

151

00:40:10.240 --> 00:40:30.460

Jisun An: that makes it even more expensive to train the model with the Rl, so people basically wanted to find a way to remove this burdens from the Rl. And that's the where the Dpo came. So they wanted to basically remove the reward model and also remove the Rl, not using the rl.

152

00:40:30.950 --> 00:40:42.090

Jisun An: so, as the name say itself, so can we fine tune a model directly using a preference data set without? Rl, what's the basic idea where the Dpo came out?

153

00:40:42.200 --> 00:40:54.974

Jisun An: So why are you avoiding? Rl, as I just mentioned? Rl, usually very unstable and computationally very, very expensive, and it's also hard to pinpoint because it it has many, many steps, many, many

154

00:40:55.510 --> 00:41:13.879

Jisun An: rollouts, or the state and the action pairs, and then sometimes, if we see output is very, very long, then the reward will be back. I mean, propagate into all this intermediary path. But we didn't know where which point was actually helping to how much. So sometimes that's also hard to do

155

00:41:13.880 --> 00:41:37.029

Jisun An: and basically to train. Basically, we need to generate multiple samples for training themselves. And that's also very costly and also training a reliable reward model is also challenging. And and those small reward difference can cause instability as well. So basically, while somehow the open AI was successful using the Etf

156

00:41:37.030 --> 00:41:43.527

Jisun An: to train their model. But it's only possible with the a lot of resources and the

157

00:41:43.990 --> 00:41:54.259

Jisun An: money as well. So basically, from the research side, I think they were looking for a solution that can be tuned the preference without such burdens.

158

00:41:55.040 --> 00:42:03.489

Jisun An: So this Dpo is also called as a preference tuning. So if you ever have have heard about preference tuning, and this may mean the Dpo.

159

00:42:03.710 --> 00:42:33.219

Jisun An: So now I will go straight to the Dpo. Loss function, because I think I will probably skip most of the derivation part. So the idea is start from the the Rl. Hf. Objective function where it has a reverse and the divergence Kr. Divergence. And the the simple idea is that because the preference we know that this can be represented by this Bradley Terry model their preference probability. So they basically do a lot of mathematical manipulation, and then they just

160

00:42:34.700 --> 00:42:36.769

Jisun An: got this loss function.

161

00:42:37.988 --> 00:42:43.411

Jisun An: Once again I will. I I have some slide, but I will, I will probably skip it.

162

00:42:44.260 --> 00:42:54.992

Jisun An: But then here you, you may be be able to noticing something. So these are loss functions. So we are trying to minimize it. But

163

00:42:56.960 --> 00:42:58.200

Jisun An: but like

164

00:42:58.420 --> 00:43:16.990

Jisun An: what it literally says, like is the maximizing the probability for the good responses and minimizing the probability for the bad responses. And then I mean, just don't change into much from the model. So these are some of the terms that that you may see from these loss functions.

165

00:43:20.230 --> 00:43:24.339

Jisun An: So so what what they did was from the

166

00:43:24.817 --> 00:43:31.760

Jisun An: I will briefly show some derivatives, but I mentioned it as an optional, because I don't want you to like

167

00:43:31.890 --> 00:43:34.269

Jisun An: learn, I mean, understand everything. But

168

00:43:34.900 --> 00:43:47.920

Jisun An: I think it's also quite interesting derivation. And it's not that difficult. So maybe if you are interested in just having a look at them, so they they start from this. The Rl Hf objective.

169

00:43:47.920 --> 00:44:12.870

Jisun An: And then they, because this Kl divergence can be simply represented as this, which is the current policy divided by the reference policy and and taking the log. So these were the definition of the Kl divergence, and then we can also take change this form by taking the negative, and then and then this form, and then they are now defining

170

00:44:12.870 --> 00:44:29.950

Jisun An: a new partition function, meaning the partition function, meaning that we have assuming we have another new policy model. And then we given given that, we we also, I mean, as we define a new policy function. This formula can be rewritten in this way.

171

00:44:31.460 --> 00:44:32.900

Jisun An: And then.

172

00:44:33.160 --> 00:44:41.570

Jisun An: Now that we we given that we have this new policy, that the the form that we had

173

00:44:42.480 --> 00:44:45.111

Jisun An: here now can be.

174

00:44:47.020 --> 00:45:09.928

Jisun An: This is now once again. These are the Kl divergence form, because it is no log like the the current policy divided by this, our new defined policy. And because we want to minimize these particular. And this is can be represented as a Kl divergence between the current policy and the new policy that we define. And because we want to minimize this, and then

175

00:45:10.270 --> 00:45:27.969

Jisun An: the this Kl divergence is minimized when these 2 policy are exactly the same. So now we know that the policy and the new policy, when they are the same, this value will be minimized. So we can have this particular function and this function turning into

176

00:45:27.990 --> 00:45:36.650

Jisun An: making that to represent our reward function in this way. So if we are now get this reward function back to the

177

00:45:36.770 --> 00:45:49.770

Jisun An: our original the the formula, then it will kind of have this duration. And now that's the where we get this predatory preference model again. So now.

178

00:45:49.890 --> 00:46:14.139

Jisun An: because we can present to the reward function, using our policy only policy models. We can also replace this preference probability using this particular function. So if you are just injecting this function, the reward function into all these 3 reward function. And then if you just organizing, and then you will see this particular loss function.

179

00:46:14.390 --> 00:46:25.320

Jisun An: And that will be this final laws that that we will gonna see. So in a way that doesn't it look quite similar to some something that you've seen before?

180

00:46:27.056 --> 00:46:50.973

Jisun An: We were kind of briefly went through. But these are basically the sigmoid a minus B. So this is something that we've seen from the when we were talking about the reward model and the the loss of the reward model because it was using the the Bradley very preference model. The form will be quite similar. But what what it says is that it can be turning into

181

00:46:51.500 --> 00:46:59.050

Jisun An: There's a simple loss. Function. as depicted here.

182

00:47:00.920 --> 00:47:03.450

Jisun An: So what if

183

00:47:03.730 --> 00:47:10.439

Jisun An: so, what we do here once again. I I know that this is a lot for for some of you, but what what we are doing here. So

184

00:47:10.510 --> 00:47:22.329

Jisun An: we want to keep the idea of using the preference data set to tune our model. But and then the open AI was using the Rl. But Rl, and also

185

00:47:22.350 --> 00:47:41.040

Jisun An: having needed to have a reward. Model itself is already expensive and challenging, so the Dpo was the idea to remove them, and they were, if we did, some mathematical manipulation, they were able to represent the reward function, using only the policy models

186

00:47:41.040 --> 00:47:56.549

Jisun An: and and those now that they can. Basically the policies are once again, is a probability distribution. So now you can have consider it as a like simple loss, and then you can train or fine tune your model using this loss function.

187

00:47:57.980 --> 00:48:21.219

Jisun An: So these are the loss that you can compute for every single different examples. And then now, what this loss function tells you is that even without rl, just having the preference data set, you can compute the loss that you can just backprop. So you can now use the simple, the normal fine tuning mechanism

188

00:48:21.220 --> 00:48:31.940

Jisun An: to fine-tune your model with the preference data to accommodate the preference to learn the human preference. I mean, that's the whole idea of the Dpo, and

189

00:48:32.210 --> 00:48:43.789

Jisun An: I know that this may not be a very, very easy, but once again, just under, try to understand the the concept. So what is the differences between the Rlatf and the Dpo.

190

00:48:43.790 --> 00:49:12.670

Jisun An: So once again the Rlhtf. So we they have a preference data, and then they use the maximum likelihood. So like no more fine tuning to get the reward model, and they use this reward model to label the reward also the sample to create the design policy using the reinforcement learning and the Dpo basically remove this this in the intermediary parts of the reward model and the reinforcement learning, then they just simply

191

00:49:13.110 --> 00:49:24.649

Jisun An: proposed a method that just from the preference data set just fine tune, the model just using the normal maximum likelihood fine-tuning. So that's the what Dpo does

192

00:49:25.740 --> 00:49:31.423

Jisun An: so if you were in the lab. 2, where we did a fine tuning.

193

00:49:32.070 --> 00:49:55.830

Jisun An: we did the the the parameter efficiency fine tuning right and very similar to it. We. The hugging face also has a Dpo library, so the exact the same code. But if you are just changing the 1 1 lines to the Dpo, then you can do the deep. You can do basically the preference tuning as well. So what it what it means is that so? The

194

00:49:55.830 --> 00:50:09.369

Jisun An: once again, the the fine tuning. You had a classification type of data set where you have a labels right? The preference data set is a slightly different. So it doesn't need a label, but it does need it need to have which one is more preferred or not.

195

00:50:09.370 --> 00:50:18.408

Jisun An: So you can still basically using the Dpo, you can train your model based on the preference data set, and maybe for some cases this would be more useful.

196

00:50:19.040 --> 00:50:20.090

Jisun An: so

197

00:50:20.520 --> 00:50:31.739

Jisun An: some of the recent open model were adapting Dpo rather than Drlacf, and they found that it was useful enough. So these are the some of the things that was going on in the Dpo.

198

00:50:33.005 --> 00:50:35.420

Jisun An: Any questions? Yeah.

199

00:50:37.700 --> 00:50:57.310

Jisun An: Right? When when did I say that? We didn't need a label?

200

00:50:57.790 --> 00:51:02.860

Jisun An: Oh, wait on who? The yeah.

201

00:51:03.950 --> 00:51:14.729

Jisun An: Oh, no, no, it doesn't need a reward model. It needs a preference data, but it it fine tune the model directly from the preference data. It doesn't need a reward model.

202

00:51:15.480 --> 00:51:33.710

Jisun An: Yeah, yeah. So both both are tuned based on the preference data. But the Rlatf required this Rl themselves and the reward model. But Dpo basically doesn't need these 2. And they can just tune based on the preference model.

203

00:51:33.860 --> 00:51:35.480

Jisun An: Yeah, that was the whole idea.

204

00:51:36.660 --> 00:51:38.520

Jisun An: Any other questions. Yes.

205

00:51:39.010 --> 00:51:44.950

Jisun An: So the reason for having is that, like they are, is quite expensive. And

206

00:51:46.260 --> 00:51:48.680

Jisun An: and then we need to train all the new models in order.

207

00:51:49.530 --> 00:52:08.929

Jisun An: And also it's unstable. But in the last place, we have discussed a little bit on deep sea, so they didn't use rl, for very much less cost right? Right? Exactly so it it dpo right before the deep sea. Dpo was still hot.

208

00:52:08.930 --> 00:52:35.750

Jisun An: I think, after this, now, Dpo is okay. There was one of such such approach that were improving or or changing to apply. Okay, once again, if you don't have resources and the money to do the Rhf, then you can go with the Dpo. So I think that's a still good option. But at the same time one downside of the Dpo. It loses the benefits of the Rl. As I, as we discussed in the earlier part of the last lecture, Rl. Had this

209

00:52:36.210 --> 00:52:58.890

Jisun An: possibility to find a optimal path that even human doesn't know to be better at for some test. So if you are not doing rl, you are actually ignoring or neglecting those opportunities. But they found that this is more or better for the reasonings. But if you are just concerning about the preference, then maybe Dpo would be just good enough.

210

00:52:59.070 --> 00:53:08.490

Jisun An: Deepstick is more about reasoning, not about preference. That's that's what I thought. And I will go to that right away. Is there any other question?

211

00:53:11.940 --> 00:53:14.649

Jisun An: Okay? Yeah. So

212

00:53:15.420 --> 00:53:35.270

Jisun An: and then we have now group relative policy optimization, which is a new Rl technique that is proposed by the dipsic. So once again, Dipsy is also long paper, so probably have a look at it. I think you should have some level of knowledge to understand. Most of the parts of these

213

00:53:35.270 --> 00:53:48.219

Jisun An: paper is not very difficult to read, so if you have some time and interested in actually go and just read it. I think it has a lot of information, and I also don't cover many parts. I only cover some parts so.

214

00:53:48.740 --> 00:54:09.729

Jisun An: And one of the key point and most surprising aspects was that they were using this grpo a new kind of method for new Rl. Method. But then this method itself was also proposed from this other paper, this thing Math Prior, than the deep sick r. 1,

215

00:54:10.570 --> 00:54:27.079

Jisun An: and here they were only focused on like mathematical reasoning. So they were using this Rl. To improve the reasonings of the mathematics problems. But so this is the the original paper that this method was proposed. So if you want to know details and also check this paper.

216

00:54:27.210 --> 00:54:50.909

Jisun An: But so so what is the Grpo? So to to say that I will go back a little bit about the Rl. In the leakage model. So for those who are a little confused by it, so the goal of the Rl. Is usually have. This policy, which is in our case, is a neural network to improve or finding the optimal policy. And this is basically the probability distribution. So at given any States.

217

00:54:51.198 --> 00:55:04.179

Jisun An: If you are injecting the State as an input to this policy, then you will give the the most probable actions, and eventually this will be the distribution of the actions, and then you will. You will select the most probable actions out of it.

218

00:55:04.180 --> 00:55:22.859

Jisun An: So in the language modeling the state would be any produced words. But then, assuming our 1st state, is, we randomly select one of the words, and that's our 1st state. So let's say that we selected the student as a as a 1st word from this current probability.

219

00:55:23.520 --> 00:55:43.409

Jisun An: then this state go into the policy as an input and as an output, it will have a list, the distribution of these actions of all the possible tokens and their probability. So out of this probability. Okay, we did a sampling. And then, so this would be now assuming that we selected the opened from these actions.

220

00:55:43.940 --> 00:56:11.090

Jisun An: So now we have this sequence of the state and the action. So this is called as an episode in the Rl. Term, and this is also called as a rolling over because it's roll over. So after this 1st step, we have State 0, which was the student, and then the action 0 will be open because we were selecting the opened. So in the next state. Now our next state will be student opened because we were selecting the opened.

221

00:56:11.090 --> 00:56:27.940

Jisun An: Now this is itself is a state, and then. Now this state will go into our policy, and then our once again given this state, our distribution over the actions will be looking like this. So now we will have a slightly higher probability for a or N,

222

00:56:28.400 --> 00:56:43.689

Jisun An: and and there so once again, we randomly sample, let's say that we were selecting there. So this selecting once again the action and the now students open, there is, became now our state, and then we are so we are doing this again over and over.

223

00:56:43.970 --> 00:57:08.600

Jisun An: assuming that we got the books, and then we can do this more like there could be n different state and the action pairs. And then, at the end of this episode, the important thing is we will get the reward. So in the language modeling or preference data, the reward was like the preference whether it was more preferred or not. But it could be any reward. So if we are talking about like method problems, then it'll be whether

224

00:57:08.600 --> 00:57:24.289

Jisun An: the output got the answer correct or not? Right? So if they got the method problem correct and the reward could be higher if it's less correct than not. And now the idea is using this reward to update and improve our policy network. So that's the what rl, does.

225

00:57:25.260 --> 00:57:52.430

Jisun An: And so here the pi, a given S is basically the probability. Choose a particular action when we are at particular state. S, so that's the our policy. And we are because this our policy network, usually a neural network where it has the parameters. We are also adding this term parameter to the pi a given s comma theta. So this means that our policy networks are parameter types

226

00:57:52.430 --> 00:57:56.809

Jisun An: so that we can tune the parameters of the police network. Given our rewards.

227

00:57:58.090 --> 00:57:58.930

Jisun An: So

228

00:57:59.200 --> 00:58:22.929

Jisun An: the reinforce algorithm is something that initial proposed a long time ago. And this is very simple idea. So basically, we want to improve this data meaning that update our policy network to be better to find the best policy models. And so, assuming that we observe a 1 particular episode of all the rollout and get the reward.

229

00:58:22.930 --> 00:58:40.199

Jisun An: then the idea is okay. If the R reward is large, then we tweak all these parameters to make that particular action at more likely. So we are increasing the probability of pi a t given St. So given at that

230

00:58:40.200 --> 00:58:46.679

Jisun An: state, and given that that for that state, that action that probably should be encouraging

231

00:58:47.590 --> 00:58:54.179

Jisun An: so, and or or if the reward was low, then basically, we are kind of rewarding that as well.

232

00:58:54.430 --> 00:59:23.550

Jisun An: Then, in this example, we knew that after students open their book, we we got the large L, then it means that our last action, and also all the previous. We will also update for the previous actions. But the last action was choosing books right? So the probability of like even students open there, the probability of the books now should be boosted. So let's say that we are increasing like 3 points. So this is what the reinforced algorithm

233

00:59:23.550 --> 00:59:26.819

Jisun An: would do to tune their policy networks.

234

00:59:27.110 --> 00:59:50.969

Jisun An: But instead of because I mean normally, the probability we cannot do this kind of summation, because if we are adding 3 more points, then it'll be over 100, so doesn't make sense. So instead of just the probability we are taking log so that we just using the log value. And basically, once we have that log probability, then this can be considered as something that

235

00:59:51.150 --> 01:00:18.040

Jisun An: we can consider their gradients. So the gradients of the log of this pi will tell the direction of the how the parameter should be tuned. Given these states and given this reverse, so basically, we will know that. Okay, for the book we need to update or like boost that particular values. And then so these are only the direction, because it's the gradients.

236

01:00:18.040 --> 01:00:29.559

Jisun An: One extra thing that we need to add is the how much we want to boost, and that can be determined by the R. Themselves. So the reward will be. Now the amount of how much we want to boost that particular action.

237

01:00:29.830 --> 01:00:37.879

Jisun An: So that's the basically and using this, we are just updating the new parameters. So this was the reinforcement reinforced algorithm

238

01:00:39.240 --> 01:00:40.360

Jisun An: any question.

239

01:00:44.380 --> 01:00:45.365

Jisun An: And

240

01:00:46.530 --> 01:01:12.050

Jisun An: so do you think this would work for like for the rl, I mean, this is something that already like there, as a Rl. Method. So it would work. Quite work. But the problem is that rl, is very noisy, and you will. So even over, like the training, the reinforcement are reinforce algorithm will be very, very slow at optimizing the the goals, because

241

01:01:12.050 --> 01:01:35.039

Jisun An: the assumption of using just the reward for updating all the parameters at the end of the actions episode. It's just it's just very noisy. The reward can be coming from something else, and there could be different factors that are relating to the reward. So so basically, I mean, these are just a mark of example. But so even though the reinforced algorithm, the idea itself is makes sense.

242

01:01:35.456 --> 01:01:50.149

Jisun An: It usually takes very, very long time to train and very, very hard to train. Rl, well, and the Grpo had a minor trick on the reinforced algorithm. And they found that this is far more better at training.

243

01:01:50.730 --> 01:01:55.557

Jisun An: So what is the idea of the grpo is that

244

01:01:56.980 --> 01:02:18.879

Jisun An: So here, if the r, if basically, you are updating this policy network for thinking about updating this policy network for every single example, then you are getting one r, so R. Was 10. Do you think that is our 10 is good or bad? I mean, it's very hard to know whether the R itself is how good the r is.

245

01:02:18.880 --> 01:02:46.049

Jisun An: So the core idea of the grpo is that the reward size R is relative. It's not something that you can interpret as the sheer value. But it should be the relative. So instead of updating this policy, based on example, episode by episode episode, here would be one sentence in our example because we are doing language, modeling not one sentence, but one prompt and output pair.

246

01:02:46.300 --> 01:02:53.710

Jisun An: So instead of updating this something at that level, they were doing this. So for the same state.

247

01:02:53.710 --> 01:03:17.204

Jisun An: they sample multiple times. So here our state 0 was starting with a student, right? And then you are sampling multiple outputs, and there could be something. Students open their book. Students are not playing. Students were quiet. So you are just sampling and creating generating different outputs. And for different output. You are getting also the reward from the reward model, as we did before, and then

248

01:03:17.640 --> 01:03:35.060

Jisun An: And then, if we are just writing it as an episode, you can kind of okay. For the 1st example, we have this observation, second and 3, rd so and then for and then. Now for each of these sample when we are updating our this policy rather than using the R value itself

249

01:03:35.060 --> 01:03:57.270

Jisun An: instead of the r, we are using something called advantage. And here the advantage is defined by the standardized value of this reward. So rather than using the R value itself, we are using r 1, which is the 10 basically in our example, minus the average of the reward value of that group

250

01:03:57.410 --> 01:04:24.089

Jisun An: divided by the standard deviation of that for you. So basically, if out of this group. So in this example, we have only 3 outputs. But there could be like 100 groups that you can sample together. And out of this 100 example those who are in the middle value, they will have 0 as advantage, and if that's the better than the average, then it will be like more than one. And if it's less than it's less than 0.

251

01:04:25.150 --> 01:04:39.620

Jisun An: Yeah. So so these advantage value is now just relative across this group of outputs, or the samples that they had from the example. So these sampling is coming from like

252

01:04:39.960 --> 01:04:59.019

Jisun An: the current policy or the reference policy I don't actually remember. But so you can use the policy network itself to generate these outputs and then get the rewards and then using the relative rewards and then update the new policy based on that. And that was the whole idea of the Grpo.

253

01:04:59.870 --> 01:05:04.200

Jisun An: Does that make sense any questions?

254

01:05:06.110 --> 01:05:20.243

Jisun An: Idea itself is quite simple. So I was talking with my colleague, and he was like, it's so simple. And why no one ever thought about this was like surprising simple, but it was very, very effective. So once again, so grpo,

255

01:05:20.590 --> 01:05:32.030

Jisun An: to update the policy you are you? You are using the reward to give the intensity of how much update that you need to do to the policy. So that was the original method.

256

01:05:32.030 --> 01:05:55.830

Jisun An: And the idea in the Grpo is that okay? So let's take the reward R as a relative value, and to make it as a relative rather than you are using one sample, you are using multiple samples so that you can compare with the others to know the how much this R value is good or bad, and then you are using this advantage value, which is the normalized value for the reward and then using it for this update function.

257

01:05:56.440 --> 01:05:59.830

Jisun An: That was a grp, then

258

01:06:01.900 --> 01:06:15.289

Jisun An: this is the their actual objective function given where this A you will see a here is the advantage. So instead of the r they have it a here, and it's the same. And and it's a still

259

01:06:15.440 --> 01:06:19.720

Jisun An: quite, I mean now I think it would be kind of getting familiar.

260

01:06:22.125 --> 01:06:43.924

Jisun An: there are some terms that would be familiar to you. So this last term is the Kr divergence. So they are still keeping some parts from the Rlhf. And they just modifying something. So this last term is the the same thing that we don't want to deviate from the reference model. And here these are. So if you are looking at just the denominator, then

261

01:06:44.440 --> 01:07:12.629

Jisun An: in terms of the pi. So you want to still want to maximize the the reward itself or the probability itself, but it's divided by the probability of the reference model. So these are something called as a surrogate objective function in it means that it's considering the momentum. So basically you want to boost the updates if it is in the same direction, and if it's the opposite direction that you want to give the lesser boost. So they. They just divided this term.

262

01:07:12.810 --> 01:07:26.820

Jisun An: And the A is the advantage which is a reward. And there is another kind of technique that are used in the Ppo, which is the clipping. And this is something that you are just preventing the excessive job. So if they found that this

263

01:07:26.920 --> 01:07:51.529

Jisun An: new value, so so the clipping is basically you are setting a particular range. And then if this jump is outside of this range, then you just don't go, that you just take the minimum or the maximum value of this range, so that you you will not update like too much. So that's the clipping. So these are the 3 basic terms that that was used. And then you see that they basically run this for each of the groups.

264

01:07:52.124 --> 01:07:57.470

Jisun An: and then get the average of it. So that's the like, the objective function of the deep state.

265

01:07:59.640 --> 01:08:20.220

Jisun An: You kind of feel that you kind of understand the right. It's not that difficult to really like. I mean, obviously, to train extra train. This model will be very, very hard, and there will be many engineering and efforts. But in terms of the theory. I think it was very easy to understand. Compared, I mean, then I expected, and

266

01:08:21.130 --> 01:08:42.880

Jisun An: and a few last thing is so. The Dipsic training pipeline once again, if you're interested in, then read the paper. But so they started from this dipsic, W. 3 0 based 3 3 base. And this was the pre-trained model. So, like other pre-trained model, they use a lot of Internet data to train this model

267

01:08:43.710 --> 01:08:44.890

Jisun An: and then

268

01:08:47.620 --> 01:09:11.660

Jisun An: and then without. So if you remember, back like, when we talked about the Openai model, we they do. They did an instruction tuned right? But they did an experiment. Okay, can we skip that instruction tuned? And can we just using this rl, to see how much they can do better or good at reasoning? So they did, just using this grpo with the particular reward. But even here, reward

269

01:09:11.660 --> 01:09:35.329

Jisun An: they didn't use any fancy preference reward, but because they were focusing on the reasoning. So they were using reasoning data set, meaning that they have verifiable answers. So it's correct. I mean, they know that whether the answer was correct or not so. They use that accuracy as their reward, and they also have a little format reward. So I think they have, like the thinking kind of format. So these were the only 2

270

01:09:35.330 --> 01:09:47.690

Jisun An: rule-based reward. So they didn't needed any other model to have defined the reward. But they simply use the rule-based reverse, and then they want the Rl. Based on this deep sick W. 3 base.

271

01:09:48.000 --> 01:09:50.510

Jisun An: That was the dipstick. R. 1 0.

272

01:09:50.620 --> 01:10:10.700

Jisun An: And this was the the graph that I showed you last term. So this was the accuracy for this the math problems aime accuracy, and as you see that as it trained more and more the accuracy was getting like higher and higher. So that was the this deep stick. R, 1 0 model.

273

01:10:11.060 --> 01:10:21.129

Jisun An: So this district R, 1, 0, basically has no instruction, tuning it, only trained, based on the Rl. And it still excelled in reasoning task.

274

01:10:24.990 --> 01:10:42.459

Jisun An: Sorry. But the problem is that as the paper mentioned, so they develops unexpected and powerful reasoning behavior. But it has faced several issues like poor readability and the language missing. So that was something very interesting that they have found. So.

275

01:10:42.510 --> 01:11:09.130

Jisun An: even though they are able to get the math problem answer correct. But if the if you are looking at their reasoning well, it was not basically human, interpretable. And once again we talked about this Rls possibility where there could be some optimal path that reach to the correct answer. But this path may not be something we would prefer, it would be preferred by the model. But maybe we are not. And that was some evidence that this dipstick was kind of showing

276

01:11:09.130 --> 01:11:26.729

Jisun An: so, and the language mixing is that it's either they are switching from English to Chinese to Korean, or English and Spanish, or mixing them words to word. So this was really not human, readable, and those so to to do that, I mean, they needed to do a little bit of

277

01:11:26.800 --> 01:11:48.979

Jisun An: instruction, tuning or supervised learning. So what did they? Was they, of course, start with the long cot data. So they created the chain of thought data, meaning that they were writing how the language model should think. So. They wrote this and create this data, a few 1,000 of those samples, and they fine tune this base model with this cot data.

278

01:11:49.100 --> 01:11:59.809

Jisun An: and after that they edit. They run through the Rl. Similar to what they did before using extra. Some reward like cot language or consistency so extra reward.

279

01:12:00.610 --> 01:12:14.609

Jisun An: And after that, using this model, they have this intermediate model, DC. B, 3 base plus this, and then using this, they generated a reasoning data. So this is like generated data from this data itself.

280

01:12:14.740 --> 01:12:35.280

Jisun An: And then they also had got the collected and gathered some other non reasoning, related data, like other other types of different prompts, more using more diverse, prompt and non. And it could be also, including like helpfulness, harmonies, etc, etc, some some other data. And using these 2 data set, they fine tuned again.

281

01:12:35.410 --> 01:12:45.850

Jisun An: and then they run another ll, and then that get into the deep Sync. r. 1. So I don't know whether you had a chance to use the dipsy r. 1, but it's the

282

01:12:46.080 --> 01:12:57.780

Jisun An: the model that is comparable to the open AI's 0 1 models, and once again I mean, they were talking about the cost. But people also say that

283

01:12:57.920 --> 01:13:17.180

Jisun An: for the dipstick to build their own facility is already a lot of money, so I think it is not a fair kind of comparison. But I think something interesting was that this work showed that the possibility of the Rl. And how it can be used in the language, modeling, etc. And also the grpo was also an interesting idea.

284

01:13:17.310 --> 01:13:32.300

Jisun An: but still to be able to usable by the human. It requires some kind of instruction tuning, but it may, but when the instruction tuning is needed will be different. So that was something that this paper suggested

285

01:13:33.440 --> 01:13:36.200

Jisun An: any question before my last slide.

286

01:13:36.520 --> 01:13:37.320

Jisun An: Yes.

287

01:13:38.008 --> 01:13:43.049

Jisun An: Did you have a heavy parameter for G, like how many.

288

01:13:44.220 --> 01:13:48.740

Jisun An: how many samples or groups we take for the reward and all allocation.

289

01:13:48.930 --> 01:14:11.860

Jisun An: Right? I I think I think, for any parameters. It requires more research to come and see what is the generator. But I think in this case, I actually don't recall what was the exact number. But I will check and then let you know. But probably like somewhere between 100 to 1,000. Yeah.

290

01:14:12.250 --> 01:14:13.339

Jisun An: But I don't know yet.

291

01:14:13.660 --> 01:14:15.729

Jisun An: I'm also very

292

01:14:15.790 --> 01:14:29.330

Jisun An: reasoning language teaching. It doesn't really matter like, even if it thinks in a different language like, suppose like, if I'm bilingual, I can think in different languages right? It's not just the one language.

293

01:14:29.360 --> 01:14:48.020

Jisun An: I don't know. I mean, that's the what what the model shows. Right? So it doesn't need to be one language. So language mixing was actually helping for them to be correct. But just Dipsy eventually wanted to provide a service to the human. I mean, even though they are not. They seem that monetization is not their prior purpose, but they still want to do it. So

294

01:14:48.090 --> 01:14:59.449

Jisun An: so I think they just needed to do. But for if we you just wanted to get better accuracy. Maybe maybe just using the Rl. Would be better. And that's the exact slide that I want to show you next. But any other question.

295

01:15:01.850 --> 01:15:11.289

Jisun An: Then I will finish, and then I will get more questions. So this is the plot coming from the Arthago paper. We talked about briefly about the Arfago before, and

296

01:15:12.112 --> 01:15:29.387

Jisun An: what this one shows is the blue line is the reinforcement learning the the purple one is the supervised learning, and the dot line is the Alphago, I mean the the leads etors, and the yellow rating is basically how power, how popular they are in the play goal.

297

01:15:29.990 --> 01:15:33.839

Jisun An: so what you say is that so initially.

298

01:15:34.240 --> 01:15:38.959

Jisun An: basically, the supervised learning was never be able to better than the human.

299

01:15:39.060 --> 01:16:05.990

Jisun An: And then what they observe is that the reinforcement. Learning based models were, I mean, slowly started, and then at some point, they were able to overcome the performance of the human. So this was the one of the 1st observation that the possibility of the Rl. And I think the Dipsig and Openai as well, and maybe some other reasoning models are what they are all doing is now see the

300

01:16:05.990 --> 01:16:19.029

Jisun An: possibility of the Rl. And they really boost the ability of the Rl. To there may be the super AI. One day the super AI will be there, and if that day comes from, then probably it would be via the rl.

301

01:16:19.810 --> 01:16:45.560

Jisun An: so so that's the last slide, and surprisingly, I won't made it. Oh, yay! And then even though we are not able to do practice with Yara, because I mean once again, it will cost a bit of computing resources and also taking a lot of time. But I thought learning R would be quite important, because there will be more research coming relating to this. So understanding some basics about

302

01:16:45.560 --> 01:17:01.689

Jisun An: what are the techniques that used in the language area will be quite important. So that's the reason that I'm pushing a little bit so hopefully, this was helpful. And yeah, let me know if you have any other questions, and I will see you next Tuesday.

303

01:17:01.820 --> 01:17:03.040

Jisun An: Thank you.