Symbol Grounding and Task Learning from Imperfect Corrections

Mattias Appelgren
University of Edinburgh
mattias.appelgren@ed.ac.uk

Alex Lascarides
University of Edinburgh
alex@inf.ed.ac.uk

Abstract

This paper describes a method for learning from a teacher's potentially unreliable corrective feedback in an interactive task learning setting. The graphical model uses discourse coherence to jointly learn symbol grounding, domain concepts and valid plans. Our experiments show that the agent learns its domain-level task in spite of the teacher's mistakes.

1 Introduction

Interactive Task Learning (ITL) aims to develop agents that can learn arbitrary new tasks through a combination of their own actions in the environment and an ongoing interaction with a teacher (see Laird et al. (2017) for a survey). Because the agent continues to learn after deployment, ITL allows an agent to learn in an ever changing environment in a natural manner.

One goal of ITL is to have the interactions be as natural as possible for a human teacher, and many different modes of interaction have been studied: non-verbal through demonstration or teleoperation (Argall et al., 2009), or natural language: an embodied extended dialogue between teacher and agent, like between a teacher and apprentice. Our interest lies in natural language interactions where teachers can provide instructions (She et al., 2014), describe current states (Hristov et al., 2017) and define concepts (Scheutz et al., 2017), goals (Kirk and Laird, 2019), and actions (She et al., 2014), while the agent asks clarifying questions (She and Chai, 2017) and executes instructed commands. Teachers can also use corrective feedback (Appelgren and Lascarides, 2020). These approaches all assume that the teacher offers information that is both correct and timely. However, humans are error prone, and so in this paper we study how agents can learn successfully from corrective feedback even when the teacher makes mistakes.

Appelgren and Lascarides' model exploits *discourse coherence* (Hobbs, 1985; Kehler, 2002; Asher and Lascarides, 2003): that is, constraints on how a current move relates to its context. But their models assume that the teacher follows perfectly a specific dialogue strategy: she corrects a mistake as and when the agent makes it. However, humans may fail to perceive mistakes when they occur,. They also may, as a result, utter a correction much later than when the agent made the mistake, and thanks to the teacher being confident, but wrong, about the agent's capacity to ground NL descriptions to their referents, the agent may miscalculate which salient part of the context the teacher is correcting. In this paper, we present and evaluate an ITL model that copes with such errors.

In §2, we use prior work to motivate the task we tackle, as described in §3. We present our ITL model in §4 and §5, focusing on coping with situations where the teacher makes mistakes of the type we just described. We show in §6 that by making the model separate the appearance that the teacher's utterance coherently connects to the agent's latest action with the chance that it is *not* so connected, our agent can still learn its domain-level task effectively.

2 Background

Interactive Task learning (ITL) exploits interaction to support autonomous decision making during planning (Laird et al., 2017). Similar to Kirk and Laird (2019), our aim is to provide the agent with information about goals, actions, and concepts that allow it to construct a formal representation of the decision problem, which can thereafter be solved with standard decision making algorithms. Rather than teaching a specific sequence of actions (as in e.g., Nicolescu and Mataric (2001); She et al. (2014)), the teacher provides the infor-

mation needed to infer a valid plan for a goal in a range of specific situations. In this work we focus on learning goals, which express constraints on a final state. The agent learns these goals by receiving *corrective* dialogue moves that highlight an aspect of the goal which the agent has violated (Appelgren and Lascarides, 2020).

Natural language (NL) can make ITL more data efficient than non-verbal demonstration alone: even simple yes/no feedback can be used to learn a reward function (Knox and Stone, 2009) or to trigger specific algorithms for improving behaviour (Nicolescu and Mataric, 2003). More extended NL phrases must map to semantic representations or logical forms that support inference (eg, Wang et al., 2016; Zettlemoyer and Collins, 2007). Like prior ITL systems, (eg, Forbes et al., 2015; Lauria et al., 2002) we assume our agent can analyse sentential syntax, restricting the possible logical forms to a finite set. But disambiguated syntax does not resolve semantic scope ambiguities or lexical senses (Copestake et al., 1999), and so the agent must use context to identify which logical form matches the speaker's intended meaning.

Recovering from misunderstandings has been addressed in dialogue systems (eg, Skantze, 2007), and ITL systems cope with incorrect estimates of denotations of NL descriptions (eg, Part and Lemon, 2019). Here, we address new sources of misunderstanding that stem quite naturally from the teacher attempting, but failing, to abide by a particular dialogue strategy: ie, to correct the agent's mistakes as and when they're made. This can lead to the learner misinterpreting the teacher's silence (silence might *not* mean the latest action was correct) or misinterpreting which action is being corrected (it might be an earlier action than the agent's latest one). We propose a model that copes with this uncertainty.

3 Task

In our task an agent must build towers in a blocks world. The agent begins knowing two PDDL action descriptions: $put(x, y)$ for putting an object x on another y; and $unstack(x, y)$ for removing an object x from another object y and placing x back on the table. Further, it knows the initial state consists of 10 individual blocks that are clear (i.e., nothing on them) and on the table, and that the goal G contains the fact that all the 10 blocks must be in a tower.

Figure 1: The colours of objects fit into different colour terms. Each individual hue is generated from a Gaussian distribution, with mean and variance selected to produce hues described by the chosen colour term. There are high level categories like "red" and "green" and more specific ones like "maroon". This figure shows examples of hues generated in each category, including one that is both red and maroon.

However, putting the blocks in a tower is only a partial description of the true planning problem, and the agent lacks vital knowledge about the problem in the following ways. First, the true goal G includes further constraints (e.g., that each red block must be on a blue block) and the agent is unaware of which such constraints are truly in the goal. Further, and perhaps more fundamentally, the agent is also unaware of the colour terms used to define the constraints. I.e. the word "red" is not a part of the agent's natural language vocabulary, and so the agent does not know what "red" means or what particular set of RGB values the word denotes. Instead, the agent can only observe the RGB value of an object and must learn to recognise the colour through interaction with the teacher, and in particular the corrective dialogue moves that the teacher utters.

The possible goal constraints are represented in equations (1–2), where C_1 and C_2 are colour terms; e.g., "red" (r for short) and "blue" (b).

$$r_1^{c_1,c_2} = \forall x.c_1(x) \rightarrow \exists y.c_2(y) \wedge on(x, y) \quad (1)$$

$$r_2^{c_1,c_2} = \forall y.c_2(y) \rightarrow \exists x.c_1(x) \wedge on(x, y) \quad (2)$$

In words, $r_1^{r,b}$ expresses that every red block must be on a blue block; $r_2^{r,b}$ that every blue block should have a red one on it. These rules constrain the final tower, but thanks to the available actions, if a constraint is violated by a put action then it remains violated in all subsequent states unless that put action is undone by $unstack$.

In our experiments (see §6), a simulated teacher observes the agent attempting to build the tower,

and when the agent executes an action that breaks one or more of the rules in the goal G, the teacher provides NL feedback—e.g., "no, red blocks should be on blue blocks". The feedback corrects the agent's action and provides an explanation as to why it was incorrect. However, linguistic syntax makes the sentence *ambiguous* between two rules—"red blocks should be on blue blocks" could mean $r_1^{r,b}$ or $r_2^{r,b}$. Thus, the agent must disambiguate the teacher's message while simultaneously learning to ground new terms in the embodied environment, in this example the terms "red" and "blue". This latter task amounts to learning which RGB values are members of which colour concepts (see Figure 1).

4 Coherence

To learn from the teacher's feedback the agent reasons about how an utterance is coherent. In discourse each utterance must connect to a previous part of the discourse through a coherence relation, and the discourse relation which connects the two informs us what the contribution adds to the discourse. In our multimodal discourse each of the teacher's utterances u connect to one of the agent's actions a through the discourse relation "correction". The semantics of correction stipulate that the content of the correction is true and negates some part of the corrected action (Asher and Lascarides, 2003). In our domain, this means that the teacher will utter u if the agent's latest action a violates the rule that she intended u to express. If $u =$ "no, red blocks should be on blue blocks" then, as previously stated, this is ambiguous between is $r_1^{r,b}$ and $r_2^{r,b}$. So, a must violate one of these two rules:

$$CC(a,u) \leftrightarrow (r_1^{r,b} \in G \wedge V(r_1^{r,b},a)) \vee (r_2^{r,b} \in G \wedge V(r_2^{r,b},a)) \quad (3)$$

where $CC(a,u)$ represents that u coherently corrects action a, G is the (true) goal, and $V(r_1^{r,b},a)$ represents that a violated $r_1^{r,b}$ (similarly for $V(r_2^{r,b},a)$). Since the semantics of correction is satisfied only if the correction is true, the rule the speaker intended to express must also be part of the true goal G; that is why (3) features $r_1^{r,b} \in G$ (and $r_2^{r,b} \in G$) in the two disjuncts.

There are two ways in which these rules can be violated. Either directly or indirectly. For $r_1^{r,b}$ the rule requires every red block to be on a blue block, therefore it is directly violated by action $a = put(o_1, o_2)$ if o_1 is red and o_2 is not blue (illustrated in S_1 of Figure 2):

$$V_D(r_1^{r,b},a) \leftrightarrow red(o_1) \wedge \neg blue(o_2) \quad (4)$$

The rule $r_2^{r,b}$ requires all blue blocks to have red blocks on them, meaning that S_1 in Figure 2 does not directly violate the rule, but S_2 does because it is only violated when a blue block does not have a red block on it:

$$V_D(r_2^{r,b},a) \leftrightarrow \neg red(o_1) \wedge blue(o_2) \quad (5)$$

So $r_1^{r,b}$ is not directly violated in S_2 and $r_2^{r,b}$ is not directly violated in S_1 but it would still be impossible to complete a rule compliant tower without undoing the progress that has been made on tower building. This is because the block which is currently not in the tower cannot be placed into the current tower in a rule compliant manner. For $r_1^{r,b}$ in S_2 the red block needs a blue block to be placed on, but no such blue block exists. Similarly, for $r_2^{r,b}$ in S_1 the blue block needs a red block to place on it, but no additional red blocks are available. In this way the rules are Indirectly violated in these states, which occurs when the number of available blocks of each colour makes it impossible to place all of those blocks:

$$V_I(r_1^{r,b},a) \leftrightarrow \neg red(o_1) \wedge blue(o_2) \; |\{o_3 : red(o_3) \wedge on(o_3,table)\}| > |\{o_4 : blue(o_4) \wedge on(o_4,table)\}| \quad (6)$$

$$V_I(r_2^{r,b},a) \leftrightarrow red(o_1) \wedge \neg blue(o_2) \wedge |\{o_3 : blue(o_3) \wedge on(o_3,table)\}| > |\{o_4 : blue(o_4) \wedge on(o_4,table)\}| \quad (7)$$

Our teacher signals if the error is due to a direct violation V_D by pointing at the tower or an indirect violation V_I by pointing at a block which cannot be placed in the tower any more (e.g., the blue block in S_1 or the red block in S_2).

When the agent observes the teacher say $u =$ "no, put red blocks on blue blocks" it can make inferences about the world, with confidence in those inferences depending on its current knowledge. For example, if it knows with confidence which blocks are "red" or "blue", then it can infer via equations (4–7) which of the rules the teacher intended to convey. Alternatively, if the agent knows which rule was violated then the agent can infer the colour of the blocks. We use this in §5 to learn the task. However, if the agent is completely ignorant about

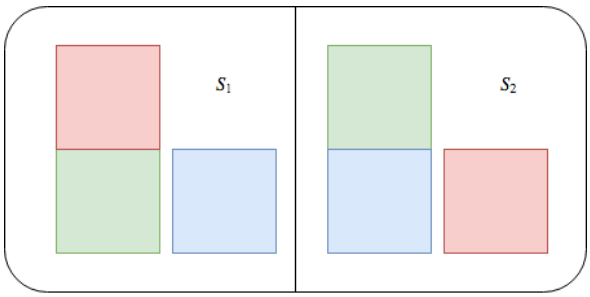

Figure 2: These two states would both be corrected if either $r_1^{(r,b)}$ or $r_2^{(r,b)}$ were in the goal.

the referents of the colour terms, it may not be able to make an inference at all. In this case it will ask for help by asking the colour of one of the blocks: "is the top block red?". Either answer to this question is suffiicent for the agent to disambiguate the intended message and also gain training exemplars (both positive and negative) for grounding the relevant colour terms.

If the teacher's dialogue strategy is to always correct an action which violates a rule (either directly or indirectly) directly after this incorrect action is executed, then the teacher's silence implies that the latest executed action does not violate a rule. This means that if the agent knows, for example, that if a green block is placed on a blue block then either green blocks must always be placed on blue blocks ($r_1^{g,b}$) or no rule constraining green blocks exists. In this way the teacher's silence implies assent.

4.1 Faulty Teacher

We've laid out the what it means for something to be coherent, assuming that the teacher always acts in the most optimal way, correcting any action which violates a rule as soon as that action is performed. However, in general a human teacher will be unlikely to perfectly follow this strategy. Despite this, an agent would still have to attempt to learn from the teacher's utterances even though some of those utterances may not fit with the agent's expectations and understanding of coherence. In this paper we introduce two types of errors the teacher can make: (a) she fails to utter a correction when the latest action a violates a rule; and (b) she utters a correction when the most recent action does not violate a rule (perhaps because she notices a previous action she should have corrected). We think of (b) as *adding* a correction at the 'wrong' time.

Since a rule can violate an action in two ways—either Directly or Indirectly—teacher errors of type (a) and (b) lead to four kinds of 'imperfect' dialogue moves:

1. Missing Direct Violations (MD)
2. Adding Direct Violations (AD)
3. Missing Indirect Violations (MI)
4. Adding Indirect Violations (AI)

In our experiments we control in what way the teacher is faulty by assigning a probability with which the teacher performs each type of mistake, e.g. P_{MD} represents the probability that the teacher misses a direct violation. Controlling these probabilities allows us to create different types of faulty teachers.

Due to the teacher's faultiness the agent must now reason about whether or not it should update its knowledge of the world given a teacher utterance or silence. In the following section we describe how we deal with this by creating graphical models which capture the semantics of coherence as laid out in this section.

5 System Description

An agent for learning from correction to perform the task described in §3 must be able to update its knowledge given the corrective feedback and then use that updated knowledge to select and execute a plan. The system we have built consists of two main parts: Action Selection and Correction Handling.

5.1 Action Selection

To generate a valid plan, the agent uses the MetricFF symbolic planner (Hoffmann and Nebel, 2001; Hoffmann, 2003)). It requires as input a representation of the current state, the goal, and the action descriptions (here, `put(x, y)` and `unstack(x, y)`. The agent knows the position of objects, including which blocks are on each other, and it knows that the goal is to build a tower. However, the agent begins unaware of predicate symbols such as `red` and `blue` and ignorant of the rules $r \in G$ that constrain the completed tower.

The aim of our system is to learn to recognise the colours—and so estimate the current state S^*—and to identify the correct goal G, given the evidence $\mathbf{X}$ which it has observed so far. We describe how shortly. The agent uses its current knowledge to construct S^* and G which are given as input to the planner to find a valid plan. Due to errors in

the grounding models or goal estimate, this may fail: eg, if the agent estimates $r_1^{r,b} \in G$ but there are more red blocks than blue blocks in S^*, making it impossible to place all of the red blocks. In such cases, the agent recovers by searching in the probabilistic neighbourhood of S^* for alternatives from which a valid plan for achieving G can be constructed (Appelgren and Lascarides, 2020). The agent executes each action in its plan until it's completed or the teacher gives corrective feedback. The latter triggers the Correction Handling system (see §5.2).

5.1.1 Grounding Models

The grounding models construct a representation of the current state S^* by predicting the colour of blocks, given their visual features. Binary classifiers represent the probability of an object being a particular colour, e.g. $P(Red_x|F_x)$ where F_x are the visual features of object x. We use binary classifiers over a categorical distribution for *colour* since the set of possible colours is unknown and colours aren't all mutually exclusive (e.g., maroon and red). We estimate the the probability using Bayes Rule:

$$P(Red_x)|F_x) = \frac{P(F_x|Red_x)P(Red_x)}{\sum_{i\in\{0,1\}} P(F_x|Red_x = i)P(Red_x = i)} \quad (8)$$

For $P(F_x|Red_x = 0)$ we use a uniform distribution—we expect colours that are not red to be distributed over the entire spectrum. $P(F_x|Red_x = 1)$ is estimated with weighted Kernel Density Estimation (wKDE). wKDE is a non-parametric model that puts a kernel around every known data point $\{(w_1, F_{x_1}), ...(w_m, F_{x_m})\}$ (where w_i are weights) and calculates the probability of a new data point via a normalised weighted sum of the values of the kernels at that point. With kernel φ (we use a diagonal Gaussian kernel), this becomes:

$$P(F_x|Red_x = 1) = \frac{1}{\sum_{i=1}^{m} w_i} \sum_{i=1}^{m} w_i \cdot \varphi(F_x - F_{x_i}) \quad (9)$$

The pairs (w_i, F_{x_i}) are generated by the Correction Handling system (see §5.2).

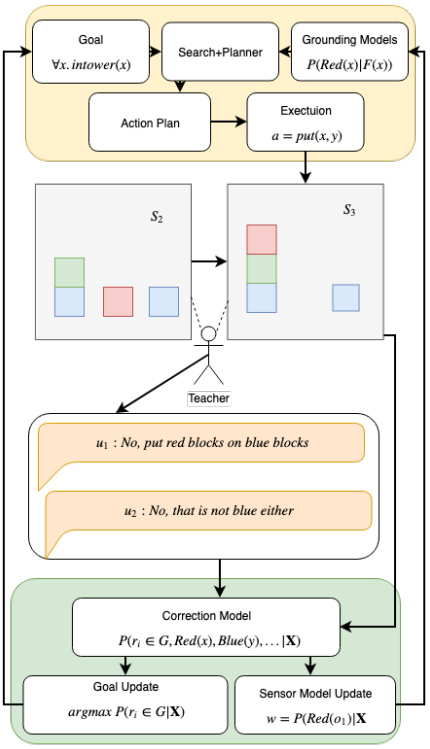

Figure 3: The agent consists of an Action Selection system (yellow) and a Learning System (green). The former uses a symbolic planner to find a plan given the most likely goal and symbol grounding. The latter uses coherence to learn the goal and grounding.

5.1.2 The Goal

In order to estimate G the agent begins with the (correct) knowledge that it must place all blocks in a tower. However, it must use the teacher's feedback $\mathbf{X}$ to find the most likely set of additional rules which are also conjuncts in G (see §5.2):

$$G = \arg\max_{r_1,\ldots,r_n} P(r_1 \in G, \ldots, r_n \in G|\mathbf{X}) \quad (10)$$

$\mathcal{R} = \{r_1 \ldots r_n\}$ is the set of possible rules that the agent is currently aware of, as determined by the colour terms it's aware of (so $\mathcal{R}$ gets larger during learning). For each $r \in \mathcal{R}$, the agent tracks its probabilistic belief that $r \in G$. Due to the belief that any one rule being in the goal is unlikely, the priors for all $r \in G$ are low: $P(r \in G) = 0.1$. And due to the independence assumption (11), the goal G is constructed by adding $r \in \mathcal{R}$ as a conjunct iff $P(r \in G|\mathbf{X}) > 0.5$.

$$P(r \in G, r' \in G|\mathbf{X}) = P(r \in G|\mathbf{X})P(r' \in G|\mathbf{X}) \quad (11)$$

5.2 Handling Corrections

When the teacher corrects the agent by uttering, for example, $u =$ "no, red blocks should be on blue

blocks" the agent must update its knowledge of the world in two ways: it must update its beliefs about what rules are in the goal, as described in §5.1.2 and it must update its models for grounding colour terms. To perform these inferences the agent builds a probabilistic model which allows it to perform these two inference. For the goal it uses the inference in equation (10). To learn the colours it performs this inference:

$$w = P(Red(o_1)|\mathbf{X}) \tag{12}$$

And adds the data point $(w, F(o_1))$ to its grounding model for red objects.

We base our graphical model on the model presented in Appelgren and Lascarides (2020) which we extend to deal with the fact that the teacher's utterance may be faulty. The model is a Bayes Net consisting of a number off different factors which are multiplied together to produce the final output probability. The model from Appelgren and Lascarides (2020) is shown in Figure 4. Grey nodes are observable while white nodes are latent. Arrows show conditional dependence between nodes. If the teacher is faultless then the agent observes that a coherent correction occurred: $CC(a, u)$. The factor for this in the graphical model:

$$P(CC(a,u)|r_1^{r,b} \in G, V(r_1^{r,b}, a), \\ r_2^{r,b} \in G, V(r_2^{r,b}, a)) \tag{13}$$

captures equation (3), which stipulates that a coherent correction occurs when a rule which is in the goal is violated. In the graphical model the factor has a value of 1 any time this is true and 0 otherwise.

The violation factors $V(r_1^{r,b}, a)$ and $V(r_2^{r,b}, a)$ represent whether or not a particular rule was violated by the action a. The agent cannot observe this directly, but must instead infer this from whether or not the objects are red and blue. As such the factor:

$$P(V(r_i^{r,b}, a)|Red_{o_1}, Blue_{o_2}) \tag{14}$$

captures equation (4) for $i = 1$ and (5) for $i = 2$. The value of the factor is 1 if the relevant equation holds and 0 otherwise. So, for example, when $V(r_1^{r,b}, a) = True$, $Red_{o_1} = True$, and $Blue_{o_1} = False$ the value of the factor is 1.

The remaining nodes $P(Red_{o_1}|F_{o_1})$ and $P(Blue_{o_2}|F_{o_2})$ are defined by the agent's grounding models. $P(F_{o_i})$ is a prior which is assumed to be a constant for all o_i. Finally, $P(r_i^{r,b} \in G)$ is the

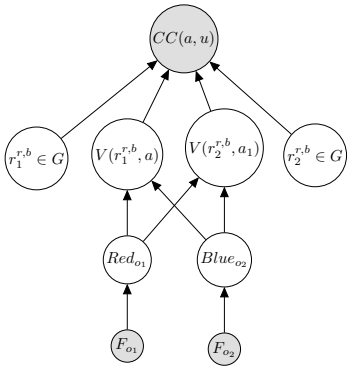

Figure 4: The nodes added to the graphical model after a correction u = "no, red blocks should be on blue blocks".

agent's prior belief that $r_i^{r,b}$ is in the goal ($i = 1, 2$). As we mentioned earlier, this is initially set to 0.1; however, the prior is updated each time the agent encounters a new planning problem instance. The prior is then set simply to the agent's current belief given the available evidence.

When the teacher designates a block o_3 on the table (thereby signaling that violation is indirect), the graphical model this generates is similar to Figure 4, save there are two additional nodes F_{o_3} and $Red_{o_3} \vee Blue_{o_3}$ (see (Appelgren and Lascarides, 2020) for details).

When the teacher stays silent the agent can make an inference which implies that no rule which is in the goal was violated. It can therefore build a model similar to Figure 4 which captures this negation of equation (3). The agent can then update its knowledge by making the same inferences when a correction occurs, but with the observed evidence being that no correction occurred. For further details on how this inference works see (Appelgren and Lascarides, 2020).

5.3 Uncertain Inferences

In this paper we assume that the teacher may make mistakes as described in §4. This introduces a novel problem for the agent since it can no longer assume that when the teacher says u that that means the utterance coherently attaches to the most recent action a. In other words, $CC(a, u)$ becomes latent, rather than observable. What *is* observable is that the teacher did in fact utter correction u immediately after action a. We capture this by adding a new (observable) factor $TeacherCorrection(a, u)$ (or $TC(a, u)$ for short) to the graphical model. When the teacher is

infallible $TC(a,u) \equiv CC(a,u)$ but not when the teacher is fallible.

The updated model is shown in Figure 5. $TC(a,u)$ is added as an observable node with $CC(a,u)$ made latent. The factor for $CC(a,u)$ still works in the same way as before, conforming to equation (3). $TC(a,u)$ imposes no semantic constraints on a or on u. However, we can use the evidence of $TC(a,u)$ to inform the agent's belief about whether $CC(a,u)$ is true or not, i.e. whether it was actually coherent to utter u in response to a. The newly added factor $P(TC(a,u)|CC(a,u)$ captures the agent's belief about how faulty the teacher is and allows the agent to therefore reason about whether $TC(a,u)$ actually means that $CC(a,u)$. In essence, it answers the question "if it is coherent to correct a with u, how likely is it that the teacher actually says u". So, if the agent believes that the teacher forgets to utter a correction with probability $p = 0.1$ then $P(TC(a,u) = False|CC(a,u) = True) = 0.1$. Or if the agent believes that the teacher will falsely correct an action which was actually correct 5% of the time then $P(TC(a,u) = True|CC(a,u) = False) = 0.05$. This allows the agent to make use of the fact that the teacher did (or didn't) utter something to still update its beliefs about which rules are in the goal and what the colour of objects are.

The agents beliefs about the teacher's fallibility could be estimated from data or could potentially be updated on the fly given the agent's observation of the interaction. However, for the purpose of the experiments in this paper we have direct access to the true probability of teacher fallibility since we explicitly set this probability ourselves. We therefore set the agent's belief about the teacher's fallibility to the true value.

The final change made to the system compared to Appelgren and Lascarides (2020) is to the way inference is done. In their paper they perform exact inference in a manner which was optimised for the structure of the graphical model and the incremental nature of the inference. However, the method relied on the fact that the majority of probability states had zero probability due to the deterministic factors in the model. When the teacher is fallible the number of zero probability states greatly falls. This leads to a situation where exact inference becomes impractical. To deal with this we deploy approximate inference, based on

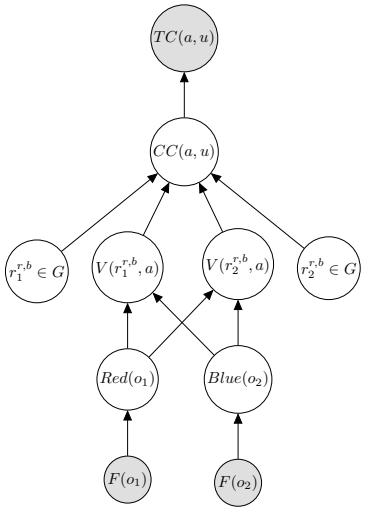

Figure 5: The nodes added to the graphical model after a correction u = "no, red blocks should be on blue blocks". Grey is observed and white latent.

a simple Bayesian Update together with a beam search method which relies on the fact that the model grows incrementally. We first find the probability for every atomic state in the newly added model chunk. This establishes a set of possible non-zero probability atomic states. These are combined with atomic states from the previous inference steps which we call the beam. The beam is the N most likely states from the previous state. Each new non-zero atomic state is combined with states from the beam if they are compatible, determined by both states having the same value for any overlapping variables. These new combined atomic states are evaluated on the full model and the N most likely are kept as a new beam, which is normalised to create a consistent probability distribution. Specific probabilities can then be calculated by summing all atomic states that match the chosen value, eg, where $Red_{o_1} = True$.

6 Experiments

In §4 we mentioned four types of teacher error and in our experiments we vary the level of the teacher's error in these different types. We believe the most likely is missing indirect errors (MI) since spotting these requires search on all possible future actions. So our first faulty teacher varies $P_{MI} \in \{0.0, 0.1, 0.25, 0.5, 1.0\}$: ie, from no errors to never correcting any indirect violations at all. Our second teacher makes mistakes with direct violations. We believe missing and adding direct violations will be linked, so we experiment

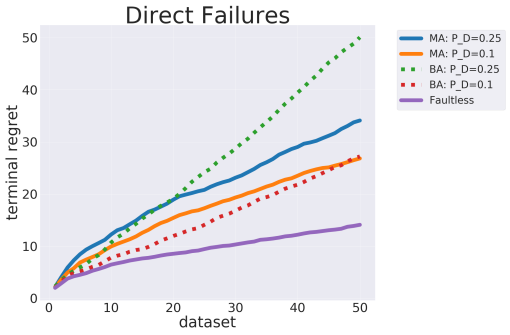

Figure 6: Cumulative regret for teachers making mistakes with direct violations. The dotted lines show the baseline agent while the solid lines show the mistake aware agent.

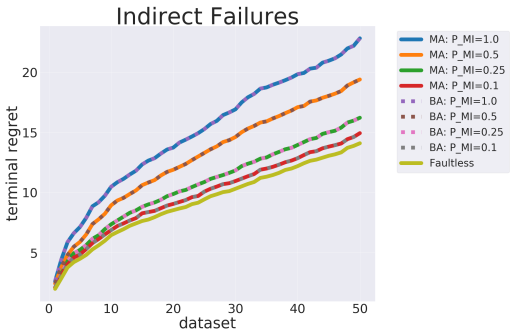

Figure 7: Cumulative regret for teachers making mistakes with indirect violations. The dotted lines show the baseline agent while the solid lines show the mistake aware agent.

with: $P_D = P_{MD} = P_{AD} \in \{0.0, 0.1, 0.25\}$. We study two types of agents in our experiments. The first, baseline agent, *BA*, ignores the fact that the teacher may be faulty. It simply uses the model described in Appelgren and Lascarides (2020). The only difference is that since the teacher is actually making mistakes, sometimes the agent may be given contradictory evidence which would cause the inference to fail. In such a situation the agent would simply ignore everything that was said in the current scenario and move on to the next planning problem instance. The second agent is a mistake-aware agent, *MA*, which makes inferences using the model from §5.3, matching its belief about the teacher's faultiness to the true probability.

In our experiments each agent is given 50 planning problems. Each planning problem has a different goal and a different set of 50 planning problem instances. The agent is reset between each planning problem, but retains knowledge between the 50 problem instances. We measure the number of mistakes the agent makes, which we call regret. A mistake is counted when an action takes a tower from a state where it is possible to complete it in a rule compliant way to one where it isn't without unstacking blocks. In Figures 6 and 7 we present the mean performance over the 50 planning problems, and we use paired t-tests to establish significance.

Let's begin by looking at the results for agents learning from teachers that fail to make corrections for indirect violations, shown in Figure 7. Clearly when the teacher is faulty the agent performs worse (a result which is shown significant through a pairwise t-test and significance threshold $p < 0.01$). However, two interesting things can be observed. First, the slope of the curves are about the same for the agents learning from the faulty teacher and those learning from the faultless teacher. What this implies is that although the agent takes longer to learn the task when the teacher misses indirect violations it does seem to reach an equal proficiency by the end. We can explain the fact that the agent makes more mistakes by the fact that it is unaware of several mistakes it is making, however, when it is made aware of a mistake it still manages to learn. The second point is that the BA and MA agents are equally good at learning at all levels of teacher error. There is a good reason for this. When the teacher misses indirect violations the agent can actually trust all other information it receives. If it is given a direct correction then it knows for certain that the teacher give a coherent correction. This is true for all the feedback the agent receives when the only error the teacher makes is missing indirect violations. For this reason there isn't actually any need to change the way in which the agent learns, which is reassuring given that we believe the indirect violations to be more likely to happen in practice.

Looking at the results when the teacher will both miss and add corrections for direct violations, shown in Figure 6, we see that the agent's performance is much worse, both compared to the faultless agent and to the agents learning from the teachers making direct violations (these results are also significant given a pairwise t-test and significance threshold $p < 0.01$). The big difference in this case is that the agent *BA* performs much worse than *MA*, especially when the likelihood of failure is higher. This is true both if we look at the final number of mistakes, but also at the slope of the

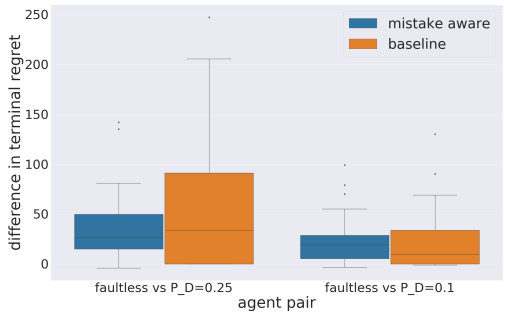

Figure 8: The difference in terminal regret when dealing with a faulty teacher vs. a faultless teacher, comparing the baseline *BA* to the mistake-aware agent *MA*.

curve, indicating that the agent is still making more mistakes by the end of training. Figure 8 shows why: it compares the difference between the terminal regret for the faultless teacher vs. the faulty one. For *BA* there is a much larger spread of outcomes, with a long tail of very high regrets. The results for *MA* reside in a much narrower region. This implies that in contrast to *MA*, *BA* performs extremely badly in a significant number of cases. The high regret scenarios can be explained by situations where the agent has failed to learn the task successfully and is therefore acting almost randomly. So, making the agent mistake-aware stabilises the learning process, allowing the agent to recover from the teacher's mistakes without completely failing to learn the task, as seen in the baseline.

7 Conclusion

In this paper we present an ITL model where the agent learns constraints and concepts in a tower building task from a teacher uttering corrections to its actions. The teacher can make mistakes, and to handle this we introduce a separation between the teacher uttering a correction (observable) vs. whether that correction coherently relates to the latest action (latent). Our experiments showed that this separation significantly reduces the proportion of situations where the agent fails to learn; without the separation, learning can go catastrophically wrong when the teacher's mistakes involve direct violations.

Acknowledgements: We thank the UKRI-funded TAS Governance Node (grant number EP/V026607/1) for their support, and three anonymous reviewers for helpful feedback. All remaining errors are our own.

References

Mattias Appelgren and A. Lascarides. 2020. Interactive task learning via embodied corrective feedback. *Auton. Agents Multi Agent Syst.*, 34:54.

Brenna Argall, Sonia Chernova, Manuela Veloso, and Brett Browning. 2009. A survey of robot learning from demonstration. *Robotics and Autonomous Systems*, 57:469–483.

Nicholas Asher and Alex Lascarides. 2003. *Logics of conversation*. Cambridge University Press.

Ann Copestake, Dan Flickinger, and Ivan A. Sag. 1999. Minimal recursion semantics: An introduction.

Maxwell Forbes, Rajesh P. N. Rao, Luke Zettlemoyer, and Maya Cakmak. 2015. Robot programming by demonstration with situated spatial language understanding. In *IEEE International Conference on Robotics and Automation, ICRA 2015, Seattle, WA, USA, 26-30 May, 2015*, pages 2014–2020.

J. R. Hobbs. 1985. On the coherence and structure of discourse. Technical Report csli-85-37, Center for the Study of Language and Information, Stanford University.

Jörg Hoffmann. 2003. The Metric-FF planning system: Translating "ignoring delete lists" to numeric state variables. 20:291–341.

Jörg Hoffmann and Bernhard Nebel. 2001. The FF planning system: Fast plan generation through heuristic search. 14:253–302.

Yordan Hristov, Svetlin Penkov, Alex Lascarides, and Subramanian Ramamoorthy. 2017. Grounding symbols in multi-modal instructions. In *Proceedings of the First Workshop on Language Grounding for Robotics, RoboNLP@ACL 2017, Vancouver, Canada, August 3, 2017*, pages 49–57.

A. Kehler. 2002. *Coherence, Reference and the Theory of Grammar*. csli Publications, Cambridge University Press.

James R. Kirk and John E. Laird. 2019. Learning hierarchical symbolic representations to support interactive task learning and knowledge transfer. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI 2019, Macao, China, August 10-16, 2019*, pages 6095–6102.

W. Bradley Knox and Peter Stone. 2009. Interactively shaping agents via human reinforcement: the TAMER framework. In *Proceedings of the 5th International Conference on Knowledge Capture (K-CAP 2009), September 1-4, 2009, Redondo Beach, California, USA*, pages 9–16.

John E. Laird, Kevin A. Gluck, John R. Anderson, Kenneth D. Forbus, Odest Chadwicke Jenkins, Christian Lebiere, Dario D. Salvucci, Matthias Scheutz, Andrea Lockerd Thomaz, J. Gregory Trafton, Robert E.

Wray, Shiwali Mohan, and James R. Kirk. 2017. Interactive task learning. *IEEE Intelligent Systems*, 32:6–21.

Stanislao Lauria, Guido Bugmann, Theocharis Kyriacou, and Ewan Klein. 2002. Mobile robot programming using natural language. *Robotics and Autonomous Systems*, 38(3-4):171–181.

Monica N. Nicolescu and Maja J. Mataric. 2001. Experience-based representation construction: learning from human and robot teachers. In *IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS 2001: Expanding the Societal Role of Robotics in the the Next Millennium, Maui, HI, USA, October 29 - November 3, 2001*, pages 740–745.

Monica N. Nicolescu and Maja J. Mataric. 2003. Natural methods for robot task learning: instructive demonstrations, generalization and practice. In *The Second International Joint Conference on Autonomous Agents & Multiagent Systems, AAMAS 2003, July 14-18, 2003, Melbourne, Victoria, Australia, Proceedings*, pages 241–248.

JL. Part and O. Lemon. 2019. Towards a robot architecture for situated lifelong object learning. In *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1854–1860. IEEE.

Matthias Scheutz, Evan A. Krause, Bradley Oosterveld, Tyler M. Frasca, and Robert Platt. 2017. Spoken instruction-based one-shot object and action learning in a cognitive robotic architecture. In *AAMAS*.

Lanbo She and Joyce Yue Chai. 2017. Interactive learning of grounded verb semantics towards human-robot communication. In *ACL*.

Lanbo She, Shaohua Yang, Yu Cheng, Yunyi Jia, Joyce Yue Chai, and Ning Xi. 2014. Back to the blocks world: Learning new actions through situated human-robot dialogue. In *SIGDIAL Conference*.

G. Skantze. 2007. *Error handling in spoken dialogue systems: Managing uncertainty, grounding and miscommunication*. Gabriel Skantze.

Sida I. Wang, Percy Liang, and Christopher D. Manning. 2016. Learning language games through interaction. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics, ACL 2016, August 7-12, 2016, Berlin, Germany, Volume 1: Long Papers*.

Luke S. Zettlemoyer and Michael Collins. 2007. Online learning of relaxed ccg grammars for parsing to logical form. In *EMNLP-CoNLL*.