Conclusion-Verified Analogical Schema Induction

Joseph A. Blass JOEBLASS@U.NORTHWESTERN.EDU
Kenneth D. Forbus FORBUS@U.NORTHWESTERN.EDU
Qualitative Reasoning Group, Northwestern University

Abstract

Learning general concepts from examples requires synthesizing shared information from those examples. The way in which the examples are compared and synthesized affects what concepts are learned through a generalization process. We present Conclusion-verified Analogical Schema Induction, a modified version of the Sequential Analogical Generalization Engine (SAGE). Conclusion-verified Analogical Schema Induction (CASI) verifies that the mapping that will be used to generalize solved cases would actually have been able to solve those same cases were one of the conclusions unknown. This increases the likelihood that generalizations learned will include conceptual information supporting the desired conclusions. We evaluate CASI on the domain of legal torts, a complex domain where cases often are highly dissimilar to each other.

1. Introduction and Background

Learning useful concepts from examples is a core area of research in Artificial Intelligence. AI researchers have undertaken an enormous variety of approaches to this task: demonstrating how to solve tasks to robotic learners (Chernova & Thomaz, 2014), learning logical definitions from instantiated relational examples (Quinlan, 1990), learning tasks from observations of human performance (Gulwani et al., 2015), learning hierarchies of concepts using Bayesian probabilistic reasoning (Grant et al., 2019), using near-misses to strengthen category boundaries (McLure et al., 2015, Rabold et al., 2022), learning event schemas from graphical event representations using neural networks (Jin et al., 2022), and more. Many of the recent advances underlying Deep Learning can be understood as using examples to learn underlying concepts (Lecun et al., 2015).

In some cases researchers might have few preconceived opinions about the concepts underlying their data and the purposes for which those concepts can be used – they toss their data into a machine learning algorithm and see what concepts the algorithms reveal (Bengio, 2012). In other cases the researchers have specific concepts in mind, and use supervised learning to learn what underlies those concepts (Krizhevsky et al., 2012). This is particularly useful when developing systems designed to reason about and solve problems: researchers might know that particular concepts are core to a problem domain, but rely on their learning algorithm to carve the boundaries of those concepts for generalized reasoning purposes (Fitzgerald et al., 2019).

This paper presents a modification of the Sequential Analogical Generalization Engine (SAGE; Kandaswamy & Forbus, 2012), a system that uses analogical reasoning to generate schemas from examples that share underlying structures. SAGE's design is inspired by psychological evidence regarding how humans learn concepts through comparison. SAGE learns concepts by positive examples, in the form of structured, relational examples, added incrementally. As usual with structure-mapping (Forbus et al., 2017), higher-order relations help indicate which lower-order relations are relevant to conclusions, and the mapping of deeper structure drives the quality of the match. But what about when the relevant higher-order structures that encode the causal relationships within a case are missing? Not every example is fully explained, when provided by an informant who assumes background knowledge, or when the example comes from observations. Another common problem in concept learning is when cases are dissimilar at the surface level or share confounding similarities. When a specific form of conclusion is available in a concept learning task, the Conclusion-Verified Analogical Schema Induction (CASI) method introduced here can be useful to overcome both these obstacles. CASI modifies how SAGE evaluates whether a mapping is a good candidate for generalization by replacing its reliance on the mapping's similarity score with a consistency check that scrutinizes the mapping. Though SAGE is a model of human analogical generalization and concept learning, CASI's modified algorithm is not presented as such a model.

We next review the structure-mapping models we use, introduce CASI, describe two experiments, and close with discussion and future work.

2. Structure Mapping

Analogy is an important tool in human conceptual learning (Christie & Gentner, 2010). Comparison of similar cases invites abstraction and helps us discern the principles underlying common cases (Gentner et al., 2009). According to Structure Mapping Theory (Gentner 1983; Gentner & Smith, 2013), analogical reasoning proceeds by first finding an alignment, or shared relational structures, between two mental descriptions of cases. Inferences are then drawn by projecting entities present in one case into the other. The Structure Mapping Engine (SME) (Forbus et al., 2017) is a computational model of analogy and similarity based on Structure Mapping Theory. SME takes in two structured, relational cases, a *base* and a *target*, and computes up to three *mappings* between them. Each mapping is composed of the set of *correspondences* between entities and expressions in the cases, *candidate inferences* projected from one case to another, and a *similarity score* that grades the quality of the match. The similarity score can be normalized according to the size of the base or target to assess how much of a case participates in a mapping.

Running SME across every case in memory would be prohibitively expensive and is cognitively implausible given the scale of human memories. MAC/FAC (Forbus et al., 1995) is a model of analogical retrieval inspired by evidence on how humans retrieve cases from memory. MAC/FAC takes a *probe*, a case like those used by SME, and a case library of other such cases. MAC/FAC uses the probe to retrieve one or more cases which are approximately most similar to the probe. MAC/FAC proceeds in two stages. First, it efficiently computes dot products between content vectors of the probe and each case in the case library. Content vectors are a compact representation of the entity types and relations of a case. This first step provides a fast, coarse measure of similarity between the probe and the cases in memory. Up to the three most similar cases are passed to the second stage, in which SME calculates mappings between each retrieved case and the probe. Again, up to three cases with their mappings are returned. At each stage fewer than three cases might be output if the second or third are much lower in score than the first.

SAGE (Kandaswamy & Forbus, 2012) performs analogical generalization. Each concept is represented by a *generalization pool* (gpool), a case library that contains generalizations and outliers (i.e. ungeneralized examples). When a case arrives, SAGE uses MAC/FAC, with the new case as the probe, to retrieve the most similar cases from the gpool. Each gpool has an *assimilation threshold*. If the similarity score from the top mapping is above that threshold, the case is assimilated into what is retrieved. If not, it is added to the gpool as an outlier. Outliers might subsequently be generalized with future cases. SAGE can also make use of analogy control predicates that require correspondences between expressions or entities in the cases, or that require a particular expression to participate in the mapping (without specifying the expression to which it must correspond).

SAGE generalizations include all facts from the cases underlying the generalization. Corresponding facts are combined, with non-identical corresponding entities replaced by abstract entities. Each fact is assigned a probability that reflects the frequency with which it was present in the assimilated cases. For example, a fact with probability 1/3 was present in one third of cases assimilated into that generalization, while a fact with probability 1 was present in all the cases and participated in each of the mappings. These probabilities are updated each time a new case is assimilated. Finally, when reasoning with generalizations SAGE uses a *probability cutoff*, such that only facts above a certain probability are put into the generalization's case for SME mappings. That means that facts idiosyncratic to individual cases are discarded when doing analogical reasoning against the generalization of those cases. Thus, as the number of assimilated cases grows, the generalization becomes more representative of its constituents' shared structures.

3. Conclusion-Verified Analogical Schema Induction

SAGE's basic function relies on SME's computation of similarity to discern the proper basis around which to build generalizations. SME generates the mapping between the new case and the case retrieved by MAC/FAC, and if the mapping score is sufficiently high, the mapping is used to assimilate the new case into the existing generalization (or to form a new generalization if MAC/FAC retrieved an outlier). Analogy control predicates allow a certain amount of fine-tuned control over the mapping generated, for example by indicating that specific expressions in the base and target cases must map to each other, or that some expression in the case(s) should be mapped. There are circumstances, however, where SME may be led astray by distractor facts shared across cases. Furthermore, SME's similarity score is driven by shared higher-order structures within cases, i.e., statements that take other statements as arguments, such as causal relationships and constraining relations. When those higher-order structures are missing from the case representations, SME may assign cases a similarity score that is so low as not to be useful as a metric for determining whether cases should be assimilated. However, just because cases do not share the higher-order structures that are the most significant contributor to SME's similarity score does not mean that no good schemas can be formed from such cases, only that SME may be unable to detect where those good schemas might come from. Under those circumstances, if SAGE is to learn useful information it will require alternative criteria for when a mapping should be used for generalization.

Imagine one wants to learn about a principle and has a set of cases illustrating it. Crucially, the principle is unknown: the facts relevant to the principle are assumed to be present in the case, but the higher order structure tying some of those facts to the conclusions are missing. Each case has an outcome, and it is understood that the outcome is derived from the operation of the unknown principle over other facts of the case. The goal is to isolate the facts consistently associated with the outcomes, even if the specific principles that explain why those facts and outcomes are associated are unknown. By identifying those patterns of facts across a series of cases, the facts associated with particular conclusions – the facts that illustrate the unknown principle – can highlighted. The goal is to build generalizations that will allow the system to understand future cases relying upon that principle, cases whose conclusions – unlike those of the training cases – are unknown.

The training cases, though potentially missing the explanation linking the facts to the outcomes, all have the information of what outcome was associated with which set of facts. We call these

Table 1. Two cases from our dataset (case text simplified from original)

	Case 1: *Trout v Bank of Belleville*	Case 2: *Conklin v. Newman*
Case Text	The defendant is a bank. The bank has a parking lot. The parking lot's exit was blocked with a chain. The chain was hard to see at night. The plaintiff was riding his motorcycle at night. The plaintiff drove into the bank's parking lot. The plaintiff collided with the chain. The plaintiff died.	The plaintiff's father owned land. A fence divided the land into two pieces. The plaintiff lived on one piece of the land. The defendant lived on the other piece of the land. The fence was on both pieces of the land. Then the plaintiff's father gave her her piece of the land. Then the defendant destroyed the fence. Then the plaintiff's father died.
Predicate Logic Representations (extracted by CNLU)	`(deathOf die68193 plaintiff67673)` `(defendants Trout_v_Bank_of_Belleville defendant66716)` `(degreeOfDifficulty see67334 (HighAmountFn LevelOfDifficulty))` `(doneBy block67045 chain67121)` `(doneBy have66867 defendant66716)` `(driverActor ride67699 plaintiff67673)` `(eventOccursAt ride67699 night67454)` `(instrument-Generic block67045 chain67121)` `(instrument-Generic collide68054 chain67121)` `(isa block67045 ObstructionEvent)` `(isa chain67121 Chain)` `(isa collide68054 Collision)` `(isa defendant66716 BankOrganization)` `(isa defendant66716 Defendant)` `(isa die68193 Dying)` `(isa drive67891 TransportInvolvingADriver)` `(isa exit66971 Exit)` `(isa have66867 Possession)` `(isa motorcycle67743 Motorcycle)` `(isa night67454 Night)` `(isa parking-lot66896 ParkingLot)` `(isa plaintiff67673 Plaintiff)` `(isa ride67699 TransportInvolvingADriver)` `(isa see67334 VisualPerception)` `(isa Trout_v_Bank_of_Belleville CourtCase)` `(objectActedOn block67045 exit66971)` `(objectActedOn collide68054 chain67121)` `(objectActedOn have66867 parking-lot66896)` `(objectFoundInLocation plaintiff67673 parking-lot66896)` `(occursDuring see67334 night67454)` `(performedBy drive67891 plaintiff67673)` `(plaintiffs Trout_v_Bank_of_Belleville plaintiff67673)` `(possessiveRelation defendant66716 parking-lot66896)` `(possessiveRelation parking-lot66896 exit66971)` `(possessiveRelation plaintiff67673 motorcycle67743)` `(primaryObjectMoving collide68054 plaintiff67673)` `(to-Generic drive67891 parking-lot66896)` `(topicOfIndividual see67334 chain67121)` `(transporter ride67699 motorcycle67743)`	`(cardinality group-of-piece14864 2)` `(comesFrom-Generic piece15433 land14512)` `(deathOf die16855 father14430)` `(defendants Conklin_v_Newman defendant15339)` `(doneBy destroy17908 defendant15339)` `(doneBy divide14674 fence14618)` `(doneBy own14480 father14430)` `(elementOf piece14864 group-of-piece14864)` `(elementOf piece15433 group-of-piece14864)` `(eventOccursAt destroy17908 piece14864)` `(eventOccursAt destroy17908 piece15433)` `(father plaintiff14417 father14430)` `(fromAgent give18445 father14430)` `(isa Conklin_v_Newman CourtCase)` `(isa defendant15339 Defendant)` `(isa destroy17908 DestructionEvent)` `(isa destroy17908 (CausingFn DamageOutcome))` `(isa die16855 Dying)` `(isa divide14674 Parcellation)` `(isa father14430 HumanFather)` `(isa fence14618 Fence)` `(isa give18445 GivingSomething)` `(isa group-of-piece14864 SetOfTypeFn SeparatePhysicalPartOfObject))` `(isa land14512 RealEstate)` `(isa live15010 ResidingSomewhere)` `(isa live15347 ResidingSomewhere)` `(isa own14480 OwningSomething)` `(isa piece14864 SeparatePhysicalPartOfObject)` `(isa piece15433 SeparatePhysicalPartOfObject)` `(isa plaintiff14417 Plaintiff)` `(objectActedOn destroy17908 fence14618)` `(objectActedOn divide14674 land14512)` `(objectActedOn own14480 land14512)` `(objectGiven give18445 piece14864)` `(on-UnderspecifiedSurface fence14618 piece14864)` `(on-UnderspecifiedSurface fence14618 piece15433)` `(outputsCreated divide14674 group-of-piece14864)` `(owns plaintiff14417 piece14864)` `(parts land14512 piece14864)` `(parts land14512 piece15433)` `(plaintiffs Conklin_v_Newman plaintiff14417)` `(possessiveRelation plaintiff14417 father14430)` `(possessiveRelation plaintiff14417 piece14864)` `(residence-Role live15010 piece14864)` `(residence-Role live15347 piece15433)` `(residents-Role live15010 plaintiff14417)` `(residents-Role live15347 defendant15339)` `(startsAfterStartingOf destroy17908 live15010)` `(startsAfterStartingOf die16855 destroy17908)` `(startsAfterStartingOf give18445 live15347)` `(to-Generic divide14674 group-of-piece14864)` `(toAgent give18445 plaintiff14417)`
Conc.	`(trespassOnPropertyByAction plaintiff67673 parking-lot66896 drive67891)`	`(trespassOnPropertyByAction defendant15339 land14512 destroy17908)`

solved cases, with the particular piece of information that may be missing in future cases labeled as the *conclusion*, that can be differentiated from all other case facts. (The conclusion must have a form that can be prespecified so that CASI can recognize conclusions in the cases where they appear.) The goal is to learn from solved cases to reliably infer that conclusion when appropriate.

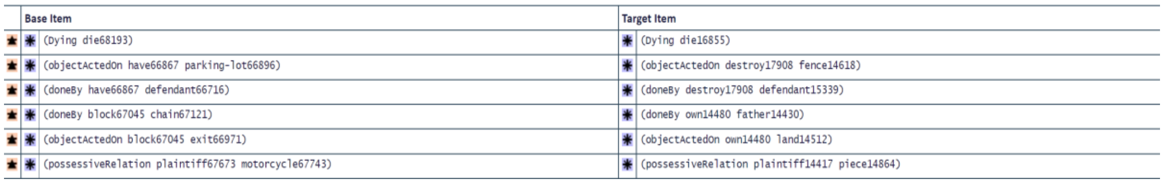
Ordinarily one would trust SAGE to learn to extract the relationships between the problem facts and the conclusions. But when the higher-order relationships between the problem facts and conclusions are not included in training cases, and especially when the cases contain distractingly similar but irrelevant facts, SAGE does not always produce reasonable answers. While there is always the risk of setting SAGE's assimilation threshold too high (leading to too few generalizations) or too low (leading to generalizations that do not capture useful information), we have found that in at least one dataset of cases there is no sweet spot where relying on SME's similarity score to control assimilation is enough to get useful generalizations. Instead, a mapping to be used for generalization must be examined to determine whether it in fact captures the information that is being used by SAGE to learn, before that mapping can be used for generalization.

To illustrate this, consider the domain of Tort Law, in which our experiments were conducted. Under the law of Trespass, a person is liable for trespassing on another's private property if the first person was on the property without permission or excuse. But imagine we do not know that principle. Instead, we are trying to build a schema of facts that illustrate the trespass rule, from cases that contain a set of facts and the conclusion of whether those facts encode a trespass. Our dataset contains 29 cases where one party trespassed on the property of another; the cases in Table 1 are taken from our dataset and are typical of the cases in it.

Unfortunately, neither of these cases contain higher-order structures, so they cannot share any such structures. The required higher-order structures connecting the conclusion facts to the rest of the facts in the case are too complex to be expressible using the current capabilities of the language system that was used to translate the cases (Blass & Forbus, 2022). Because of that lack of shared higher-order structure, SME generally scores the similarity of these cases quite low. In fact, if the assimilation threshold is set even to 0.1 (a very low score), only 3 of the 29 cases in the dataset will be assimilated together. And if it is set lower still (say, 0.01), generalizations are generated, but of poor quality and do not capture the information that defines a trespass and that is shared across many cases. There is no "sweet spot" between these scores. Why does this happen?

Examining these two cases, it should be clear to the reader that, from the perspective of trespass, the proper analogs here for the trespassers are the plaintiff in Case 1 and the defendant in Case 2, the proper analog for the property being trespassed upon are the parking lot and the land, and the trespassees should be the defendant in Case 1 and the plaintiff in Case 2. But SME gets distracted by the facts that both plaintiffs own something (in Case 1, a motorcycle, and in Case 2, a piece of land), and by the presence of death. In particular, it gets distracted by multiple statements involving a relation (like objectActedOn and possessiveRelation) and aligns things it should not align (Figure 1). Because the trespass statements are not aligned, if this mapping is used to create a generalization, the system will have learned nothing about trespass.

Perhaps all that is needed is simply to require that the trespass conclusions (the statements describing the trespass) correspond with each other? Unfortunately, that does not solve the problem



	Base Item	Target Item
★	✱ (Dying die68193)	✱ (Dying die16855)
★	✱ (objectActedOn have66867 parking-lot66896)	✱ (objectActedOn destroy17908 fence14618)
★	✱ (doneBy have66867 defendant66716)	✱ (doneBy destroy17908 defendant15339)
★	✱ (doneBy block67045 chain67121)	✱ (doneBy own14480 father14430)
★	✱ (objectActedOn block67045 exit66971)	✱ (objectActedOn own14480 land14512)
★	✱ (possessiveRelation plaintiff67673 motorcycle67743)	✱ (possessiveRelation plaintiff14417 piece14864)

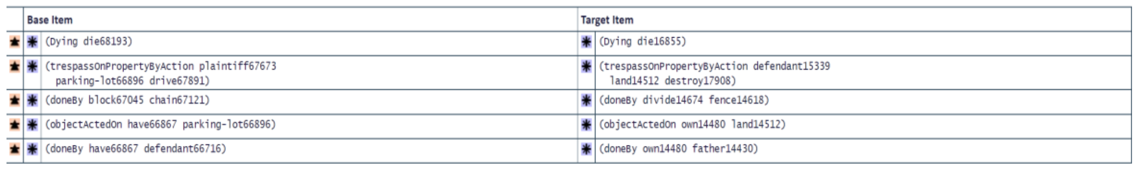
Figure 1: Default mapping between Case 1 and Case 2.

in these cases: SME will put those two facts—and therefore the entities in the specific conclusion facts—in correspondence, without properly aligning the *other* statements involving those same entities that explain why those entities play the role they do in the conclusion statement (Figure 2). And having correctly aligned the plaintiff in Case 1 with the defendant in Case 2, but incorrectly aligned the Death statements with each other, it cannot even align the role relations attached to the Death statement, since doing so would violate SME's 1:1 mapping constraint (by aligning the plaintiff in Case 1 – who is already aligned with the defendant in Case 2 – with the father in Case 2). Thus this is a mapping that puts the conclusions into correspondence, but does not provide much useful information about trespass.

In fact, while these two cases were chosen for illustration purposes because SAGE consistently assimilates them together when using the lower assimilation threshold, they actually should *not* generalize together, because SME will not generate a useful mapping from them for learning about trespass. *Trout* should assimilate with an as-yet-unseen case, while our experiments suggest *Conklin* is an outlier that should not assimilate with other cases in the dataset (at least, not without rerepresenting the facts of the case). Unfortunately, the only way to prevent SAGE from generating a generalization from them is to set the assimilation threshold sufficiently high that no other useful generalizations are made either. Thus, when similarity scores are sufficiently low because of a lack of higher-order structure shared across cases and because of shared distractor entities and relations, SME's similarity score becomes an ineffective standard by which to determine whether two cases should be assimilated. That holds true even when the relevant information (the case conclusions) are required to be mapped. In such a situation, will SAGE simply be unusable for datasets that have these qualities?

Fortunately, this is not the case. We can use a variant of SAGE to construct useful generalizations by ensuring that a task-specific conclusion predicate plays a very specific role in the mapping (described below). SME is perfectly capable of generating the mapping that both maps the conclusion and those relevant statements, but it might not get a chance to, either because a higher-scored mapping is used for generalization first, or (as we found in our experiment) the cases that *should* be usefully generalized together cannot be because they have already been assimilated with other cases in ways that occlude the useful information about those cases. In other words, in tasks where there is a known type of conclusion to be drawn when learning a concept, the algorithm can use that constraint to search for the most productive prior generalizations or outliers for assimilating a new example. That is what CASI does.

Conclusion-verified Analogical Schema Induction (CASI) is essentially a consistency check that replaces SAGE's reliance on an assimilation threshold when learning in tasks where there is a known type of conclusion to be drawn. The algorithm is presented in Figure 3. CASI works by withholding the conclusion from the probe case (step 1) when performing an analogical retrieval for generalization (step 3). It then checks whether that conclusion is among the candidate inferences from the retrieved mapping (steps 4-7). That is, it checks whether the mapping from the retrieved (solved) case to the probe's case facts *without* the conclusion would allow SME to generate the probe's withheld conclusion. The held-out conclusion thus verifies the mapping. If the mapping



		Base Item		Target Item
★	✱	(Dying die68193)	✱	(Dying die16855)
★	✱	(trespassOnPropertyByAction plaintiff67673 parking-lot66896 drive67891)	✱	(trespassOnPropertyByAction defendant15339 land14512 destroy17908)
★	✱	(doneBy block67045 chain67121)	✱	(doneBy divide14674 fence14618)
★	✱	(objectActedOn have66867 parking-lot66896)	✱	(objectActedOn own14480 land14512)
★	✱	(doneBy have66867 defendant66716)	✱	(doneBy own14480 father14430)

Figure 2: Mapping between Case 1 and Case 2 with Required Correspondence of Conclusion Fact.

```
Given case c, gpool g, conclusion predicate cp:
CASI(c, g):
  1. Probe pc = nonConclusionFacts(c)
  2. Conclusions SC = {conclusion(c)} [each of form cp(X)]
  3. Reminding r = reminding(pc, g)
    4. If r:
      5. For mapping m in r:
        6. For sc ∈ SC:
          7. If sc ∈ candidate-inferences(m):
              8. RetrievedCase retr = baseOfMapping(m)
              9. doSageGeneralizeWithMapping(c, retr, m)
          Else: [do nothing; go to next conclusion]
        Else: [do nothing; go to next mapping]
      If no mapping:
        10. CASI(c, caseLibMinus(g, retr)
        11. [perform another reminding sans retrieved cases]
    Else: [no retrieved cases, add c ungeneralized]
```

Figure 3: the CASI Algorithm.

generates the held-out conclusion, the conclusion is reintegrated into the probe case and the mapping, and SAGE uses the mapping to assimilate the probe with the retrieved case (steps 8-9). (If there is more than one conclusion statement, CASI will use the first mapping that generates one of them.) Otherwise, it examines another mapping for the same retrieval, followed by additional retrievals, (steps 10-11) until it finds one that works or runs out of candidates.

The key idea is to ensure that the mapping used to generalize example cases captures the connections between the case facts and the case conclusions, connections that might be implicit and only revealed across multiple cases governed by the same principle. This is using analogy for inductive reasoning: SME constructs a mapping that explains what is common across cases, and CASI verifies that the mapping in fact can explain that which the system is meant to learn. Thus CASI checks whether the facts SME has identified as shared can be used to project the conclusions to the cases in question. The reason this works is not because CASI necessarily forces SAGE to construct and use a good mapping, but rather that CASI will lead SAGE to *reject* assimilating cases with useless mappings, leaving those cases available to be assimilated with once the right future case (and mapping) comes along.

There is to our knowledge no evidence that humans engage in this kind of reasoning when learning concepts, though it is not implausible that they might. Humans can inspect the analogies they form between cases and the inferences they draw from those analogies, and they might well do so when they use those same analogies to learn about concepts. But CASI was developed to achieve more accurate performance on a machine learning task, not to implement a model of human cognition or make predictions about human performance. As such, we evaluate its performance against baseline SAGE and against a large-language model's performance, rather than against human data.

4. Experimental Validation

4.1 Experiment 1: Examining Generalizations

CASI was evaluated on Trespass, Assault, and Battery cases from the Illinois Intentional Tort Qualitative Dataset (Blass & Forbus, 2022). This dataset of historical Illinois cases includes the original statement of case facts, syntactically simplified statements of those facts, machine translations of the simplified facts into predicate logic using the Companions Natural Language Understanding system (Tomai & Forbus, 2009), predicate logic representations of case conclusions, and information about the cases such as the decision year, the court, the legal claim at stake, and whether the claim was successful or not. A *positive* case is one where the legal claim is found to have occurred, i.e., one of the parties trespassed; a *negative* case is one where the claim failed. Our experimental validation only used CASI to create generalizations from positive cases, because in the legal domain only positive cases should be expected to have relevant information in common. That is, positive cases are the ones that encode the events in which the claim at issue happened, and often negative cases only have in common the fact that they are not positive cases (unless there is a particularly common set of facts under which people repeatedly bring failing legal claims). While negative cases can be useful for delineating category boundaries (McLure et al., 2015), using negative cases in this way was not a part of our initial experiments for CASI. In all, CASI was tested on 29 positive cases in Trespass, 12 positive cases in Assault, and 30 positive cases in Battery.

CASI was first tested in comparison to the traditional method of using SAGE, with a match constraint to ensure that a legal case conclusion participated in the mapping. That is, the control condition involved feeding cases into SAGE as usual and letting SAGE pick the best mapping with which to generalize, provided the mapping included an expression correspondence that mapped the conclusion fact of the probe.[1] The experimental condition used the CASI algorithm as described above in Figure 3. Cases were assimilated within gpools specific to the case doctrine (i.e., there was one gpool for Trespass cases, another for Assault cases, etc.). Cases were given to both algorithms in the same order, a hand-generated ordering designed to group like cases together. This manual ordering was designed to help SAGE and CASI find the best generalizations they could, to make those generalizations as useful as possible for automated legal reasoning. (Our automated legal reasoning results are reported in Blass & Forbus (in prep), which also investigates using a random order of cases).

One of the motivations for developing CASI is that legal cases that feature the same claim may be extremely dissimilar, or similar in ways that can be distracting to SAGE. We therefore tested both algorithms using assimilation thresholds of 0.01 (a low standard of similarity reflecting the dissimilarity between cases)[2] and of 0 (relying on CASI's conclusion check in the experimental condition and the mapping constraint in the control condition rather than SME's similarity score).[3] CASI's consistency check is meant to replace reliance on SME's similarity score, and we indeed found that CASI using a threshold of 0 performed better than when using any threshold at all.

[1] We do not require that the solution statements in the probe and the retrieved case map to each other because some cases have more than one solution statement. SME's 1:1 mapping constraint prevents any statement in a case from corresponding to more than one statement in the other. When mapping a case with two solutions to a case with one, requiring solutions to correspond would involve arbitrarily deciding which solutions should correspond (or trying to break the 1:1 mapping constraint, thus producing no mapping). Requiring only that the probe's solution facts be mapped allows SME the flexibility of mapping as many such facts as it can, and if not all of them can be mapped, picking the best one(s) for the mapping.

[2] Pilot investigation found that SME assigned almost no pairs of cases a normalized similarity score of even 0.1, leading us to use a very low similarity threshold in order to produce any generalizations at all.

[3] An assimilation threshold of zero has been used to produce joint probability tables for other ML algorithms (Halstead & Forbus, 2005).

Table 2. Facts sought in each generalization.

Doc-trine	Facts Necessary to Infer Doctrine	# Necessary Facts
Tres-pass	(1) Trespassee owns property; (2) Trespasser is on property OR (a) Trespasser owns an object AND (b) the object is on the property; (3) The trespassing event is done by the trespasser OR by the trespasser's object; (4) The trespassing event brings the trespasser or the trespasser's object onto the property.	4-5
Assault	(1) A threat occurred; (2) The threat was performed by the as-saulter; (3) The threat was against the assaultee.	3
Battery	(1) A touch occurred; (2) The touch was offensive OR the touch was harmful; (3) The batterer performed the touch; (4) The vic-tim was the person touched.	4

Similarly, when using a threshold of 0, SAGE will assimilate all cases into a single useless generalization. We therefore only report results of CASI using a threshold of 0 and SAGE using a threshold of 0.01. A probability cutoff of 0.6 was used, meaning that when a case is constructed for analogical reasoning from a generalization, facts below 0.6 probability are excluded from the case. We selected a probability cutoff of 0.6 as a sweet spot, high enough to include only facts present in a majority of cases in a generalization (to isolate the facts core to a legal claim), but low enough to include facts present in two of three cases in three-case generalizations.

The two approaches were evaluated by examining the generalizations generated by each approach in each doctrine. We looked at the facts in each generalization that were above the probability cutoff (i.e., the facts SME would put into a case constructed from that generalization) to see whether they contained the facts relevant to the legal doctrine at issue. To avoid the possibility of user bias, we pre-generated the facts necessary to find the legal claim. We verified whether those facts were in a generalization, rather than make a judgment call as to whether the generalization was somehow "sufficient." The facts relevant to each doctrine are presented in Table 2. Note that our representation system uses neo-Davidsonian event representation, so the sentence "Pat walked into the house," for example, would be represented with three statements: one defining a walking

Table 3. Results

Method	Doctrine	# Genl'ns Produced	# Un-genl'd Cases	% Genl'd Cases	# Correct Genl'ns	# Partial Genl'ns	# False Genl'ns	Proport. Correct Genl'ns	Proport. Partial+ Genl'ns
CASI	Trespass	9	5	83	4	2	3	0.444	0.667
	Assault	3	3	75	2	1	0	0.667	1
	Battery	6	6	80	2	2	2	0.333	0.667
SAGE	Trespass	5	14	52	0	1	4	0	0.2
	Assault	3	2	83	2	0	1	0.667	0.667
	Battery	6	12	60	2	1	3	0.333	0.5

event, one identifying Pat as the doer of that event, and one defining the event as being into the location of the house (plus statements declaring the house to be a house, and Pat to be a human).

Generalizations were coded into three categories: a *correct* generalization contained all the necessary facts; a *partial* generalization contained all but one necessary fact; and a *false* generalization was missing more than one necessary fact. The data are reported in Table 3.

CASI outperforms SAGE on trespass cases. Indeed, SAGE was essentially unable to learn useful generalizations from trespass cases. Moreover, the proportion of generalizations that are correct *or* partially correct using CASI substantially exceeds those using SAGE. And closer examination revealed that the missing high-probability statement in many of the partial generalizations using CASI was the role relation indicating the perpetrator of the action. An examination of those generalizations revealed that these role relations often differed across cases (e.g., doneBy, performedBy, bodilyDoer, etc.). If these statements were re-represented to reflect their shared nature (i.e., they connect an action to its actor), we hypothesize that many partial CASI generalizations would become correct generalizations. On the other hand, SAGE generalizations in this domain often contained high-probability facts that had nothing to do with the outcomes of cases, for example, that the parties in both cases were walking dogs or driving a car at the time the tortious behavior occurred. Experiment 1 suggests that in this domain, CASI will synthesize more cases together and in a better way, resulting in more generalizations that will be potentially usable for reasoning by a different system, such as the one described in Blass & Forbus (in prep). Whether this is the case is the subject of Experiment 2.

4.2 Experiment 2: Using Generalizations

Constructing generalizations from cases is not done for its own sake, but so that learned concepts can be used for some task. We examined the performance of a legal reasoning system that reasons about legal cases using analogical generalizations, comparing its performance when using SAGE versus CASI generalizations. These legal reasoning techniques are described and evaluated in greater detail in Blass & Forbus (in prep, manuscript available upon request). We tested two legal reasoning systems and compared them to a large-language model baseline. Both experimental systems involve holding out a case from the dataset, constructing generalizations from the remaining positive legal cases (using either SAGE or CASI), then using the resulting generalizations to reason about the held-out case.

In Analogical Reasoning with Positive Generalizations (ARPG), the generalizations are applied to the held-out case directly by analogy. As mentioned previously, in the legal domain cases illustrating the same legal principle can be highly dissimilar to each other except for the facts that define that legal principle. When learning from such dissimilar cases, properly formed generalizations ought only to encode those facts that define the claims at issue, with facts incident to individual cases having low probabilities and therefore not participating in the case constructed from the schema for analogical reasoning.[4] ARPG relies on this idea by examining the candidate inferences generated by a mapping from the generalization to the held-out case: if there is only one candidate inference, in the form of a conclusion statement, then all other facts in the generalization—and thus core to the legal claim—participate in the mapping and therefore have a corresponding fact in the

[4] The assumption that generalization will strip away cases' idiosyncratic facts is specific to the legal domain, where concepts underlying legal doctrines are fairly abstract and can be grounded in a wide variety of different specifics. We do not assume in general that a schema-building process will strip away *all* facts incident to cases illustrating some concept. Cases contain all sorts of facts that may be correlated with a case's outcomes, and if enough of the cases in a generalization share those correlates, then the generalization will as well.

held-out case. If the held-out case contains all facts core to the legal claim (other than the legal conclusion), then the held-out case contains an instance of that legal claim being met. On the other hand, if there are candidate inferences in addition to one for the legal conclusion, then there are facts present in the generalization that are not present in the held-out case. These missing facts correspond to what legal facts would be required for the held-out case to be a positive example of the legal claim in question. ARPG thus concludes whether a legal case is positive or negative simply by counting the extra candidate inferences: if the only candidate inference is the legal conclusion, then ARPG concludes that the case is a positive instance; if there are extra inferences besides the conclusion, it concludes that the case is a negative instance.[5] (ARPG only reasons with generalizations because ungeneralized cases will contain extra facts and therefore generate many candidate inferences.)

We tested ARPG using Precision@6, meaning the system checked that it generated the correct answer in the first six mappings it tried. We did this to separate the system's ability to generate the proper answer from its ability to do so using the first mapping from its first retrieval, that is, to separate the system's reasoning ability from its ability to retrieve the right case for reasoning on the first try. (Given the low SME similarity scores when comparing cases in this dataset, the retrieval task poses its own problems and is its own area of research.) For our experimental validation we also ran several conditions varying the number of additional candidate inferences ARPG would tolerate before concluding that a case was negative. The claim that legal generalizations should encode only facts relevant to the legal claim at issue is a theoretical postulation, and the results of experiment 1 demonstrated that these generalizations do not yet perfectly encode the legal claims at issue. The generalizations often contained extra facts incidental to the legal claim that we theorize would fall away with more cases assimilated, but that did not in our experiments. Therefore we ran ARPG with a tolerance for 0, 1, and 2 extra candidate inferences, to compare its performance on SAGE and CASI generalizations. Allowing extra candidate inferences acknowledges that there may be facts incidental to a claim that participate in a generalization.

We also compared SAGE and CASI generalizations using a technique called Reasoning with Rules Learned from Generalizations (RRLG). RRLG constructs Horn clauses from generalizations and uses those rules to reason about a case using backchaining. RRLG constructs its Horn clauses by first replacing the generalized entities in the generalization with logical variables. It then extracts the conclusion statement and installs it as the Horn clause's consequent (conclusion predicates are known, so conclusion statements are identifiable), and installs the other facts of the generalization as antecedents of the Horn clause. RRLG filters out Horn clauses whose antecedents will not bind all consequent variables. RRLG's performance is evaluated by checking that it can correctly derive legal conclusions in positive cases, and correctly *fail* to derive a legal conclusion in negative cases.

We compared the performance of RRLG and ARPG to themselves when reasoning with legal schemas created using SAGE versus when using CASI. (For a comparison of these reasoning techniques relative to each other, see Blass & Forbus (in prep).) We also compared their performance to that of legalBERT, a LLM BERT model specialized on legal cases. LegalBERT was not used to develop legal schemas (or trained on those schemas); it was tested instead by turning each case into a multiple-choice question by varying the original case conclusion (reversing the parties' roles,

[5] Again, extra candidate inferences might simply correspond to additional correlate facts, and not be evidence that some concept is inapplicable. By hypothesis, the legal domain is an exception: because the facts common across legal cases in some domain operate at a fairly high level of abstraction, they are less situationally-specific and will therefore will share fewer correlates. As a corollary, any high-probability facts that survive in a sufficiently large generalization will also generally be present in the cases being reasoned about, and therefore will not be proposed as candidate inferences. Thus if a sufficiently large legal generalization includes facts other than the consequent or its antecedents, those correlative facts should generally also be in the new case being reasoned about, and so will not impede ARPG's performance.

Table 4. Results (Experiment 2). Numbers represent the number of cases correctly solved by each approach, in absolute terms and as percentages, and Precision. ARPG algorithms tested using Accuracy@6.

Technique	Method	Overall Cases	Assault Cases	Battery Cases	Trespass Cases	Positive Cases	Negative Cases	Pos Prec.	Neg Prec.
ARPG 0CIs	CASI	35 (35%)	6 (35%)	10 (25%)	19 (44%)	7 (10%)	28 (97%)	.778	.308
ARPG 0CIs	SAGE	31 (31%)	6 (35%)	12 (30%)	13 (30%)	4 (6%)	27 (93%)	.571	.290
ARPG 1CIs	CASI	47 (47%)	9 (53%)	16 (40%)	22 (51%)	19 (27%)	28 (97%)	.905	.354
ARPG 1CIs	SAGE	33 (33%)	7 (41%)	12 (30%)	14 (33%)	10 (14%)	23 (79%)	.588	.277
ARPG 2CIs	CASI	58 (58%)	9 (53%)	22 (55%)	27 (63%)	30 (42%)	28 (97%)	.938	.412
ARPG 2CIs	SAGE	36 (36%)	9 (53%)	22 (55%)	13 (30%)	10 (14%)	26 (90%)	.714	.302
RRLG	CASI	47 (47%)	8 (47%)	21 (53%)	18 (42%)	23 (33%)	24 (83%)	.793	.338
RRLG	SAGE	48 (48%)	10 (59%)	24 (60%)	14 (33%)	19 (27%)	29(100%)	.95	.363
legalBERT	-	33 (33%)	9 (53%)	14 (36%)	10 (23%)	23 (33%)	10 (35%)	.535	.175

reversing the legal outcome, and reversing both the parties and the outcome). Its performance was assessed by prompting it with the simplified text description of a case's facts and checking which answer it selected. These results are presented in Table 4.

Results were compared using proportion tests (all significance results reported at $p<0.05$). ARPG with 0 additional CIs performed statistically the same with both CASI and SAGE forming the generalizations used. However, with 1 or 2 additional CIs tolerated, ARPG using CASI generalizations performed significantly better than when using SAGE generalizations. Notably, these techniques using CASI generalizations not only performed significantly better overall, but also specifically on Trespass cases. This is consistent with the results from Experiment 1 showing that CASI made the greatest improvement in schema learning for Trespass cases.

RRLG did not perform significantly differently when using CASI generalizations than when using SAGE generalizations. However, there was a non-significant trend of RRLG performing better on Trespass cases and on Positive cases when using CASI than when using SAGE, and performing better on Assault and Battery when using SAGE than when using CASI. RRLG performed significantly better on negative cases when using SAGE than when using CASI (indeed, when using SAGE, RRLG got all the negative cases correct).

The trend in RRLG's results, though not significant, support the claim that CASI leads to better legal generalizations from this dataset. Consider RRLG's evaluation: if a case is positive, then to solve the case, RRLG must successfully fire one of the rules it learned from a generalization of other cases in the same legal doctrine. But when a case is negative, successfully solving the case means that all of RRLG's learned rules *failed* to fire. Perversely, this means that the worse RRLG's rules are at encoding some legal doctrine, the *better* it will perform in close negative cases, even as it performs worse on positive cases. Thus, the fact that RRLG using CASI's performance trends better on positive cases than when using SAGE, despite (and consistent with) its worse performance on negative cases, is an encouraging signal that CASI's generalizations are actually capturing relevant information about the legal principles governing these cases.

The observation that the quality of learned legal principles is better revealed by performance on positive cases than on negative ones holds true for ARPG as well. And in ARPG with 1 or 2 extra CIs, performance is not only significantly improved on Trespass cases when using CASI relative to SAGE, but on positive cases as well.

Even as performance was better on negative cases, precision was higher for positive cases: the systems were more likely to be correct when identifying a positive case than a negative case. For ARPG, precision was higher on both negative and positive cases when using CASI generalizations than SAGE generalizations. For RRLG, precision was higher (and performance trended better) when using SAGE generalizations.

Comparing our techniques to the baseline, RRLG when using both CASI and SAGE significantly outperformed legalBERT. ARPG when using SAGE was never able to significantly outperform legalBERT. When using CASI, ARPG with 0 extra CIs allowed did not significantly outperform legalBERT, but allowing 1 or 2 extra CIs allowed ARPG using CASI to significantly outperform legalBERT.

5. Discussion, Limitations, & Future Work

These results demonstrate that Conclusion-verified Analogical Schema Induction can be an effective tool for learning generalizations of facts underlying concepts. Our experiments demonstrate that while CASI cannot guarantee that perfect generalizations are formed from our dataset, the generalizations formed capture useful conceptual information about the domain. More importantly, where CASI had the greatest effect improving generalizations in Experiment 1 (i.e., on Trespass cases), that improvement was reflected in improved performance when reasoning with those generalizations in experiment 2.

CASI is similar to Inductive Logic Programming (Muggleton & De Raedt, 1994) in that it is a form of inductive inference that operates over symbolic representations. Both involve examples that start with irrelevant information which is stripped away. ILP always produces rules, whereas analogical generalization produces probabilistic schemas and maintains outliers, both of which are applied via analogy, rather than unification. ILP typically operates offline, in batch mode, whereas analogical generalization is incremental. ILP uses both positive and negative examples in crafting its rules, to maximize coverage of positive examples and minimize coverage of negative examples, whereas CASI currently only uses positive examples. While SAGE has been extended to incorporate automatically-derived near-misses, and thereby benefit from negative examples (McLure et al 2015), extending CASI to use near-misses is an avenue for future work.

CASI is only applicable in domains where the learning system knows ahead of time the specific form of the conclusion facts within its training data, because CASI requires removing the conclusion from a case during the first step of generalization. CASI has been shown to be useful when learning from case sets that give SME little to grasp onto in guiding generalization, for example, when causal information connecting case facts to outcomes is unknown, or in domains where cases may be dissimilar to each other or share irrelevant distractor features. Legal reasoning is such a domain because cases illustrating legal principles may be dissimilar in every way except for the specific facts directly relevant to the purportedly illegal conduct. It remains to be seen whether CASI leads to improved learning and performance in areas where SAGE is already an effective learning system.

The twin observations that CASI's performance relative to SAGE is more improved for Trespass cases than for Assault and Battery, and that CASI performs better with a threshold of 0 than 0.01, supports the conclusion that CASI is most useful for use in datasets where SME will consistently assign very low similarity scores to case comparisons. CASI's improved performance with a threshold of 0 suggests that even the vanishingly low assimilation threshold is an impediment to generalizing useful cases in this domain. Furthermore, because of the nature of the dataset and how it was generated, Assault and Battery cases contained more information specific to legal claims than did Trespass cases. That is, Assault cases consistently represent legally relevant information with the predicates `threateningAgent` and `threatenedAgent`, while legally relevant information in Battery consistently involves an entity that is a `TouchingEvent`. In contrast, the facts describing legally-relevant information in Trespass cases use role relations used to represent many different kinds of events in the cases. These common and repeated role relations make cases distractingly similar to each other even as similarity scores remained low. For datasets such as this one, SME's similarity score is ineffective at signaling to SAGE when two cases should assimilate.

CASI has a clear limitation relative to SAGE, which is in its efficiency. SAGE efficiently uses MAC/FAC and SME to find mappings, and if the mappings score high enough, produces a generalization. Using CASI requires scrutinizing each mapping's candidate inferences, and potentially going back to the well repeatedly until a mapping either produces the required conclusion or all cases in the library have been exhausted (although it would be trivial to set a maximum number of retrievals CASI could perform). This efficiency limitation is related to the fact that SAGE is a model of how humans naturally learn concepts through comparison of similar cases, one that reproduces human psychological results (Forbus et al., 2017), while CASI represents a substantially more deliberative, conclusion-oriented problem-solving approach and is not a model of everyday human reasoning. We were inspired by our domain: lawyers not only use analogies, but legal reasoning and argument is more deliberative and rigorous than the everyday learning that humans naturally engage in. Determining whether CASI in fact tracks with deliberative human cognition is one potential area of future work; another is to test CASI on more cases and more domains. It remains to be seen whether CASI will only lead to improved performance in complex domains featuring highly dissimilar cases such as legal reasoning, or whether it is a useful technique for concept learning in general.

Blass & Forbus (in prep) report detailed results of a study extending Experiment 2, using CASI generalizations to reason about new legal cases. Curious readers are directed to that work.

Acknowledgements

This research was supported by the Computational Cognition and Machine Intelligence Program of the Air Force Office of Scientific Research under award #FA9550-20-1-0091.

References

Bengio, Y. (2012). Deep learning of representations for unsupervised and transfer learning. In *Proceedings of ICML workshop on unsupervised and transfer learning* (pp. 17-36). JMLR Workshop and Conference Proceedings.

Blass, J.A., & Forbus, K.D. (2022). The Illinois Intentional Tort Qualitative Dataset. *In Proceedings of the 2022 JURIX Conference on Legal Knowledge and Information Systems.* Saarbrucken, Germany.

Blass, J.A., & Forbus, K.D. (in prep). Analogical Reasoning, Generalization, and Rule Learning for Legal Reasoning about Common Law Torts. (Manuscript available upon request.)

Chernova, S., & Thomaz, A. L. (2014). Robot learning from human teachers. *Synthesis Lectures on Artificial Intelligence and Machine Learning, 8*(3), 1-121.

Christie, S., & Gentner, D. (2010). Where hypotheses come from: Learning new relations by structural alignment. *Journal of Cognition and Development*, *11*(3), 356-373.

Fitzgerald, T., Short, E., Goel, A., & Thomaz, A. (2019). Human-guided trajectory adaptation for tool transfer. In *Proceedings of the 18th International Conference on Autonomous Agents and MultiAgent Systems* (pp. 1350-1358). Montreal, Canada.

Forbus, K. D., Ferguson, R. W., Lovett, A., & Gentner, D. (2017). Extending SME to handle large-scale cognitive modeling. *Cognitive Science*, *41*(5), 1152-1201.

Forbus, K. D., Gentner, D., & Law, K. (1995). MAC/FAC: A model of similarity-based retrieval. *Cognitive Science, 19*(2), 141-205.

Gentner, D. (1983). Structure-mapping: A theoretical framework for analogy. *Cognitive Science*, *7*(2), 155-170.

Gentner, D., Loewenstein, J., Thompson, L., & Forbus, K. D. (2009). Reviving inert knowledge: Analogical abstraction supports relational retrieval of past events. *Cognitive Science, 33*(8), 1343-1382.

Gentner, D., & Smith, L. A. (2013). Analogical Learning and Reasoning. In D. Reisberg (Ed.), *Oxford Library of Psychology* (pp. 668-681). New York, NY: Oxford University Press.

Grant, E., Peterson, J.C., & Griffiths, T. (2019). Learning deep taxonomic priors for concept learning from few positive examples. In *Proceedings of the 41st Annual Conference of the Cognitive Science Society (CogSci)* (pp. 1865-1870). Montreal, Canada

Gulwani, S., Hernandez-Orallo, J., Kitzelmann, E., Muggleton, S.H ., Schmid, U., & Zorn, B. (2015). Inductive programming meets the real world. *Communications of the ACM,* 58(11), 90-99.

Halstead, D. T., & Forbus, K. D. (2005). Transforming between propositions and features: Bridging the gap. In *Proceedings of the 20th National Conference on Artificial Intelligence (AAAI)* (pp. 777-782). Pittsburgh, PA.

Jin, X., Li, M., & Ji, H. (2022). Event Schema Induction with Double Graph Autoencoders. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies* (pp. 2013-2025). Seattle, WA.

Kandaswamy, S. and Forbus, K. (2012). Modeling Learning of Relational Abstractions via Structural Alignment. *Proceedings of the 34th Annual Conference of the Cognitive Science Society (CogSci)*. Sapporo, Japan.

Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2017). Imagenet classification with deep convolutional neural networks. *Communications of the ACM, 60*(6), 84-90.

LeCun, Y., Bengio, Y., & Hinton, G. (2015). Deep learning. *Nature, 521*(7553), 436-444.

McLure, M., Friedman, S., & Forbus, K. (2015). Extending analogical generalization with near-misses. In *Proceedings of the 29th AAAI Conference on Artificial Intelligence* (pp. 565-571). Austin, TX.

Muggleton, S. & De Raedt, L. (1994). Inductive Logic Programming: Theory and Methods. *Journal of Logic Programming 19*(20), 629-679.

Quinlan, J. R. (1990). Learning logical definitions from relations. *Machine Learning*, *5*(3), 239-266.

Rabold, J., Siebers, M., & Schmid, U. (2022). Generating contrastive explanations for inductive logic programming based on a near miss approach. *Machine Learning 111*(5), 1799-1820.

Tomai, E., & Forbus, K. D. (2009, March). EA NLU: Practical Language Understanding for Cog. Modeling. In *Proceedings of the 22nd International Florida Artificial Intelligence Research Society Conference, FLAIRS-22* (pp. 117-122). Sanibel Island, FL.