Jacob Kilver
Knowledge Based AI
Final Exam
31 July 2016

## Question 1

Version spaces is a method for learning a concept incrementally, usually from a limited number of examples. However, it differs from pure incremental concept learning in that version spaces operate on two concepts simultaneously – a general and a specific concept. As positive and negative examples are provided, these two versions of the concept towards each other. When these two concepts meet (in terms of abstraction) the concept is now learned. The primary advantages of version spaces are that convergence is guaranteed (given enough examples) and the level of abstraction at which the concept is learned is ideal, neither too general nor too specific.

In this case the “concept” is “pump is malfunctioning” and the examples are the state of the pump as measured by the sensors. The examples should not only be negative examples (malfunctions) but also positive examples, when the pump is operating normally. (If only negative examples are provided to the version space agent, only one concept will move, and the two concepts may never meet.)

The primary difficulty with using version spaces is the number of examples necessary to reach convergence. In lecture the most dimensions of any example was 4, and each of those dimensions could only take on a few values. In this case there are almost 3 times as many dimensions and the values are continuous. It could take more than 8 malfunctioning examples to converge, effectively leaving JCA without any usable model for the cause of the malfunction. This means that many more failures could happen before the root of the problem is determined.

One alternative would be to simply use incremental concept learning. While the concept that is learned after 8 malfunction examples may not be completely correct, at least there would be a usable concept that could be used to predict a malfunction.

## Question 2

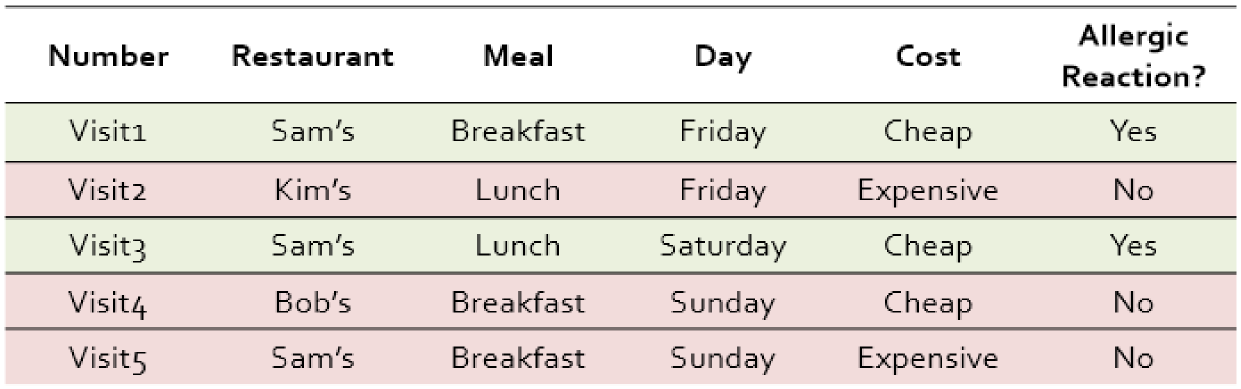
Representing knowledge is extremely difficult when ambiguity is involved, and very few things are as ambiguous as human language. In lecture we discussed ambiguity of words within a language, but in an increasingly globalized world, translating between languages is becoming increasingly common. This can lead to ambiguity when certain phrases in one language are translated literally into another. For example, assume we have an agent that only knows the grammar and dictionary of English and Spanish. Someone asks this agent to translate the English phrase “What’s up?” into Spanish. The agent uses its syntactical knowledge stored in frames to perform this translation, but the result would not make much sense to a native Spanish speaker. That is because the phrase “What’s up” is not valid in a context outside of the English language, even though Spanish has equivalent words for each of English words.

In lecture we discussed frames solely in the context of verbal knowledge. However, frames could be used to organize visual and audio information as well. However, even these can be applied out of context. Take for example a robot butler that has learned to answer the door when the doorbell rings. That evening its master is watching some television when it hears a sound similar to what it has learned as the doorbell. So the robot goes to the door and opens it to find no one there. This is because the sound of the doorbell came from the television, not the doorbell. In this case the context was more than likely the same (doorbell sound heard within a house) but the robot failed to take into account a specific part of the current context (television is on) that would have permitted the disambiguation here.

## Question 5

To solve the problem of identifying an allergy using incremental concept learning (ICL), we begin with the first example. This example can be either positive or negative, which differs from version spaces in which the first example must be a positive example. As new examples are provided, the concept is either generalized or specialized. If the example fits the current definition of the concept but is not an example of the concept, the model is specialized to exclude the example. If the example does not fit the current definition of the concept but is an example of it, the model is generalized so as to include the example. This is slightly different from the version space method which uses two models – a specific and a general model. Positive examples generalize the specific model, while negative examples specialize the general models.

See the chart below for the example of learning a food allergy using ICL. Beginning with the first row, the model of the concept is exactly this example. The next row is a negative example that is already excluded by the model, so nothing changes. The third example is a positive example that would otherwise be excluded, so the model needs generalized. The reaction occurs at Sam's restaurant with any cheap meal. The final two examples are not included in the model, so with the 5 examples below the model converges to the same result as with version spaces. Notice that this result was achieved without having to expand and prune as with version spaces. The complexity of ICL is less than that of version spaces.



| Number | Restaurant | Meal | Day | Cost | Allergic Reaction? |
|---|---|---|---|---|---|
| Visit1 | Sam's | Breakfast | Friday | Cheap | Yes |
| Visit2 | Kim's | Lunch | Friday | Expensive | No |
| Visit3 | Sam's | Lunch | Saturday | Cheap | Yes |
| Visit4 | Bob's | Breakfast | Sunday | Cheap | No |
| Visit5 | Sam's | Breakfast | Sunday | Expensive | No |

Version spaces are not able to provide labels "online." In other words, version spaces cannot provide a label until it has converged. There are two concepts at play, so how would an agent decide which concept to use when labelling? On the other hand, incremental concept learning has only one concept, so it can provide a label while the system has not yet converged. However, it is not known when ICL converges, while a version space is guaranteed to converge. If examples can be provided until convergence, version spaces is preferred, but if the number of examples is limited or if the concept needs to be used before convergence, ICL should be used.

## Question 6

Both explanation based learning and learning by correcting mistakes operate on the same knowledge representation – a hierarchical semantic network. The primary difference between these two is how this knowledge is used. Explanation based reasoning attempts to classify a new, unlabeled example given the knowledge that it has. In lecture, the knowledge structure was used to classify objects as cups. It is assumed that the knowledge the agent possesses is correct, or at least that it represents all the knowledge available to the agent at the given time. Thus this method is best suited for new situations. For example, a virtual agent that needs to identify pictures of cars. The agent has built up a definition for a car and can explain why certain pictures do or do not contain cars.

With learning by correcting mistakes, an example is provided where the classification was wrong. The correct label is not necessarily provided, merely the fact that the label provided by the agent was incorrect. In lecture, an object was classified as a cup that should not have been classified as a cup or an object that was a cup was labelled as not a cup. As such, it is known that the knowledge representation that the agent has is flawed. Something needs to be added or removed in order to correctly classify the example. This method is therefore best suited for post-processing. It necessarily needs a set of examples that were misclassified and a teacher that has labeled them as misclassified. So in the example given in above, the agent has classified a number of pictures incorrectly, and a teacher has informed the agent.

Comparing version spaces to learning by correcting mistakes, we know that they have different knowledge representations. However, in both cases labelled examples are provided to the agents which then modify their model of the concept to refine or enlarge it. In the case of version spaces, fields in the two "versions" are changed based on incoming examples. For correcting mistakes, the hierarchical network is changed based to include or remove the mislabeled example.

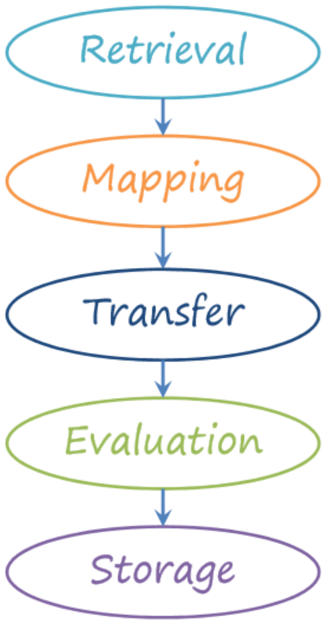
## Question 7

The steps of analogical reasoning in order are: retrieval, mapping, transfer, evaluation, and storage. As an example, take that of designing a car with low wind resistance.

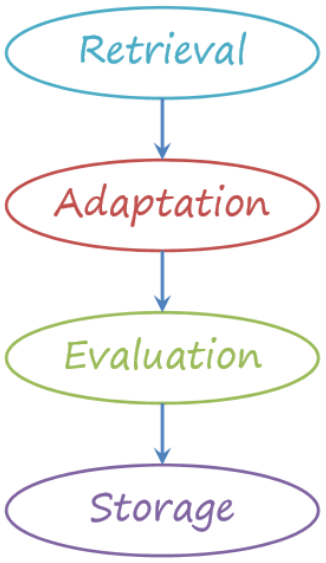
1) Retrieval – The agent knows about the wind resistance of various objects in nature. It wants to minimize the wind resistance of the car, so it selects the object with the lowest known wind resistance. For this example, assume this is the shape of a falling drop of water (teardrop).
2) Mapping – Find the higher order (deep) similarities between the teardrop and the shape of a car. In this case it is the relationship between the air and water. Since water takes the shape of its container, it is molded by the air flowing across it, resulting in an aerodynamic shape. The relationship between the air and the car needs to be the same.
3) Transfer – Take the solution from the retrieved case and apply it here. The shape of the water droplet was molded by the air flowing over it, so the car needs to have its shape molded the same way.
4) Evaluation – Determine how well the solution addresses the current problem. This would be tested in a wind tunnel to measure the wind resistance
5) Storage – Store the new problem and solution as a case for future use. This should be done even if the solution is not a good one so that poor solutions can be avoided.

The power of analogical reasoning lies in its ability to connect disparate fields, unlike other forms of knowledge based design. The only connection required for analogical reasoning is that of relationships between objects. On the other hand, for case based reasoning, the objects and relationships must be similar. Take the figure below. Case based reasoning is unable to map examples from one domain to another, while analogical reasoning can. So in this case, case based reasoning would have to retrieve designs of low-drag vehicles. It would be unable to map from the domain of nature to the domain of automobiles.

Analogical reasoning process



Case based reasoning process


Retrieval
Adaptation
Evaluation
Storage