[Music]

Welcome to the fifth lecture of the fifth week of this course on machine learning. This lecture will be about case based reasoning. Case based reasoning CBR, is the process of solving new problems based on the experiences from solutions of similar past problems expressed, in an alternative fashion CBR does solve a new problem by remembering previous similar problems and by reusing knowledge of successful problem solving for those old cases. Case based reasoning can be motivated by assumptions such as similar problems normally of similar solutions, many domains are regular in the sense of successful problem solving schemes are invariant over time. Case based reasoning should be contrasted by rule-based reasoning, an interesting analogy here is always when you look at Law systems, so there are some law systems in the world typically inspired by a German tradition which are very much controlled by very strong rule systems, while the anglo-saxon legal system is typically case based. So these are actually two camps in problem solving, probably there is no simple answer that one is better than the other, they probably complete each other in some way. However the focus today is on the case based approach, so actually cases are stored what we call in a case base or case memory could be retrieved and used. When a successful solution to the new problem is found, an adapted case can be stored in the case base to increase the competence of the system for future problem solving. So actually this functionality is what implements learning behavior in this kind of system. Technically case based reasoning is primarily supported by the techniques described in the previous lecture on similarities lending or memory based learning, particularly examples here are the schemes for distance and similarity measures which is very important also for to apply in case based reasoning. So important aspect of a CBR system is how the cases are represented. It's necessary to represent both the problem and it's solution and one can say that the representation problem here is even tougher than in in the pretty simple conceptual learning cases we looked at earlier because a case including its solution is a complex entity, so a representation needs engineering. So the representations are problem the basis for case retrievals and the representation of case solutions are the basis for case adaption. So different from the representation are the classical ones, feature value lists, graphs, predicate logic. As for all machine learning techniques feature selection is a crucial sub process here. The choice of representation is also crucial for which kind of similarity based learning techniques can be applied and utilized. One common way of modeling a CBR system cycle of activity is the following, four main phases, retrieval we try to find similar problems in the case space, reuse

the proposed solutions belonging to the similar problems, revise the solutions so they fit and can solve the current problem and then take the revised and adapted case and store it back into the case based again which we call retain in this case. Actually these goals can also be depicted a little more in detail so we can see here we have a problem we have a new case, we make a retrieval, where we search among all the old cases, we select a few that has some close similarity and we try to reuse them on the solved case that leads to some revisions and some repair, and then we make a modified copy of the case that really solves the problem at hand and then we can store that problem again in the case space to be used for further problem-solving in the future.

So let's talk shortly now about each of these four phases if we now accept this model of case based reasoning. So the first phase we call retrieve, the goal is to find a small number of cases from the case base with the highest similarity to the current case. So retrieval can be based on different kinds of the techniques, one can choose to have some kind of indexing techniques for everything in the base and of course different approaches to indexing, also you can focus more on the similarity learning techniques, instance-based learning techniques, for example apply the nearest neighbor methods using a weighted sum of features, finally there are also other approach is something called template retrieval where you return all cases that fit certain parameter settings. In the end you typically return a very small set of closest cases for further processing.

The next phase is called reuse. So reusing retrieved solution can be very simple if the storage solution can be used unchanged that's the solution for the new problem. This is not the typical case though, so otherwise adaption is required. Two maintain techniques for adaption in CBR are we can say it Transformational but actually takes the previous solution and using and modify that based on some domain specific transformation operators, so it actually transform the solution. The other approach core Derivational where you reuse not the specific solution but the algorithm methods and rules that were used to generate the original solution, and then you apply them again in a different way to produce a new solution. So you need to create something different but you can do it more in a, I would say the transformation is a more surface oriented way and derivational is more in a more deep sense.

The third phase is called revise. So in this phase feedback related to the solution constructed so far is obtained from the environment. This feedback can then be given in the form of correctness rating of the results or in the form of a manually corrected revised case. So the

retrieved case is adapted to reflect difference between the new case and the received case. If the solution is unsuccessful the retrieved case can be repaired by using the domain specific knowledge. After the solution has been successfully adapted to the target problem the result can be stored as in your case in memory. The final phase is called Retain so retained phase is the learning phase of a CBR system in the sense that is where the revised case is added to the case base again. So depending on circumstances a revised case can be either actually retained or simply forgotten that's also a possibility. The new case is integrated in the memories structure either by indexing or by being positioned in the feature space, depending on if you have a indexing philosophy or more of a similarity space or instance space philosophy for how to structure the case base. It may also be that the case base structure may be consolidated due to the addition of a new case by changing somehow the indexing mechanism or reconsidering importance of features and so on. Actually when we look back we can say that that if we talk about learning in this kind of system, there are two of the phases that are more relevant to talk about learning, on one hand I would say it's relevant to talk about learning in the revised face because there actually you create something new based on the experience from the new case, and then more technically to integrate this new thing into the existing memory structure in the retain phase, so actually learning is more related to the two later phases while problem solving are more related to the two first phases of the whole CBR cycle. It could be interesting to compare CBR to other reasoning and learning approaches, one key difference between the implicit generalizations in CBR and explicit generalization in mainstream inductive machine learning systems is concerned with when the generalizations are made. Most inductive algorithms make their explicit generalization from a set of training examples before the presence of any target problem, we call this eager generalization. This is in contrast to CBR which delays is typically implicit generalization until testing time. We call is lazy generalization. CBR therefore tends to be a good approach for rich, complex domains in which there are multitude of ways to general a case, which is non-trivial to handle in the mainstream inductive machine learning tradition.

Let's look at some advantages with CBR systems. Cases are sometimes the best way to represent knowledge, especially when the available models or theories are unreliable (weak theory domains). The good case is of an inefficient shortcut to the search for good solutions. The capability of CBR is organically increased by the incremental addition of new cases. Intrinsic advantages of an old case can be strengthened by an incremental improvements due

to the adaptions triggered by new cases. The processes of storing cases and reading cases can be computationally less expensive than alternative learning mechanisms.

Criticism of CBR is not uncommon, especially from advocates of much more straight formal methods. This critics of CBR argue that CBR is an approach of accepts anecdotal evidence as its main operating principle, so that without statistically relevant data for backing up simplest generalizations there is no guarantee that the generalizations are correct. As an example of response to this criticism recent research has defined CBR within a statistical framework that formalizes case-based inferences a specific type of probabilistic inference. So this is of course some kind of defense for the method, but still this debate goes on.

CBR is a very much applied area, so not surprisingly there are many very application active application sectors in this case. So on this slide a few of those are mentioned, the more important ones, Helpdesk and Customer service systems, Recommender systems not surprisingly, Medical applications in particular diagnosis, applications in Law in particularly in the anglo-saxon law systems as I've already commented on, technical troubleshooting and financial management and decision-making. Finally I just want to give you a very small example how cases of this kind could look like, so here you find an example from technical troubleshooting, so you can see you have a few stored cases and you have a new case and you have of course a number of features as we have in machine learning in general, and as been said here, what you need to do is you have to describe the problem in a domain-specific and clever way, so that it's naturally easy to find similar objects in a constructive way, and then of course the next problem is how should you also represent the solution which I normally is even more tricky task to design a good representations for solutions because the solutions of course should then in a later stage of the process be adapted. So there is no time unfortunately in this part of the course to go any deeper but if you are interested in CBR please study the further reading that is recommended and then you can get a more much more information about both applications and concrete cases like this. So this was the end of this lecture thanks for your attention. The next lecture - 6 will be on the topic of the tutorial on the assignments for this week. Thank you very much