NaturalLanguageProcessing-Lecture12

Instructor (Dan Jurafsky) :How about now? Oh, yeah, okay. Great, all right. So I'm Dan Jurafsky. I'm taking for Chris today as he emailed you because he had to leave town. And so we're shifting around the schedule a little bit and we're doing semantic role labeling today and he'll return to more parsing on Thursday – on Wednesday and Monday.

And most of these slides were taken from a really good tutorial that I recommend you look at. It has a lot more slides that I'm not covering by Scott and Kristina. Kristina was actually a student here, just graduated a few years ago, and some slides from Sameer, who is a student of mine. So a lot of students of ours have worked on semantic role labeling.

So let's just start with a – you can tell a slide from Kristina and Scott and imagine this sentence, "Yesterday Kristina hit Scott with a baseball." There's lots of ways that we could have expressed that sentence in different orderings or with different kinds of syntactic constructions.

And the intuition should be that if we're trying to build information extraction system or a question answerer or anything that the fact that these are syntactically different, that that have different parse trees – completely different parse trees, shouldn't blind us to the fact that they have similar meanings.

So we'd like to be able to capture this idea of shallow semantics that says that we want to know who did what to whom. That Kristina was the agent of some kind of violent act and Scott was on the receiving end and the baseball was the instrument of this violent act. I might be able to do that for any kind of parse tree of this type. Let me skip those.

And so intuitively we talk about an agent of some kind of event, the active person who does the acting, a patient or theme which is the person affected by some action, the instrument and maybe some kind of time phrases or things like that.

And our goal for today is gonna be can we extract these kind of – I call them shallow semantics because it's not full logical form of something very complicated which Chris will get to later, it's relatively shallow. Can we extract these very simply from strings of words?

And you'll see it's very similar to the kind of parsing we'll be talking about the last week. And in particular what we're gonna propose is that the way to do this is to build a parse tree of the types you've looked at and notice that certain constituents like "with a baseball" wherever they happen to appear in the parse tree can be labeled with a particular semantic form.

So we're gonna actually build a parse tree and just map the constituents in the tree to semantic form. It's a very simple process, in fact. So, some more examples of some particular semantic roles, so here we have two breaking events where just to make it very clear that the subject, the syntactic subject of the sentence is different.

In the first case the thing that's broken is the object of break, it occurs after the verb, and in the second case it's the subject of break.

So we'd like to, again, extract the same semantics, figuring out that the window was the theme of the breaking event despite the – what seems like a syntactic difference.

Similarly for an offering event we can have somebody offering something to somebody and we can order these things in all sorts of various ways. It could – the thing that's offered, the guarantee, could be after the verb, it could be the second thing after the verb, it could be before the verb, and so on.

So this is a passive sentence where it's offered could say after the passive verb. So, lots of different surface realizations, same shallow semantic form. So why do we care about this? And we should always ask that for any NLP task. A lot of NLP application – a lot of NLP methodologies and tools and the math we use are very useful inside NLP for some NLP tasks.

But at this point in the course, you should ask yourself, "What can I do with this that isn't just relevant to NLP?" So the first and the most useful thing people have found semantic roles useful for was question answering.

And so if I ask a question, like, you know, "What's the first computer system that defeated Kasparov?" And the sentence in some web page that has the answer is in a different form and my goal is to pull out just the string Deep Blue as an exact answer, then a semantic parse can help me find that and find that exact answer.

And more generally if I ask a question like, "When was Napoleon defeated?" I know that once I've found the page that has the answer I want to pull up the snippet. Let's say I'm doing snippet generation for Google and I want to pull out just the snippet that has the answer I know that I'm searching for a case where Napoleon was the patient or theme of some defeating event and that what I want to know is – never mind the notation, I'll come back to this – is the [inaudible] application.

A lot of NLP methodologies and tools and the math we use are very useful inside NLP for some NLP tasks. But at this point in the core you should ask yourself, "What can I do with this that isn't just relevant to NLP?"

So the first and most useful thing people have found semantic roles useful for was question answering. So if I ask a question, like, you know, "What's the first computer system that defeated Kasperov?" And the sentence in some web page that has the answer is in a different form and my goal is to pull out just the string Deep Blue as an exact answer, then a semantic parse can help me find that exact answer.

And more generally if I ask a question like, “When was Napoleon defeated?” I know that once I’ve found the page that has the answer I want to pull up the snippet. Let’s say I’m doing snippet generation for Google and I want to pull out just the snippet that has the answer I know that I’m searching for a case where Napoleon was the patient or theme of some defeating event and that what I want to know is – never mind the notation, I’ll come back to this – is the temporal phrase that tells me the time of this event. Okay?

So more generally any kind of who did what to whom question we can say – and this is gonna help us find exactly the answer phrase in the sentence. So the task is – the question answerer here is one more level even more specific than snippet generation for Google. Everyone knows what a snippet is, what I mean by snippet? Yes? So when I do a search I get back the short phrase with bold face in it and that’s the snippet. So we like to do better than snippet generation that’s question answerer.

Lots of other tasks. So machine translation, if I’m translating between English and let’s say Farsi where the verb is in a very different position in a sentence in Farsi than it is in English. And Farsi or Hindi or Japanese the verb comes at the end and so where the arguments occur in order is very different. So figuring that out it’ll help us to know the semantics of the arguments so I can see the moving around.

Summarization, information instruction, I won’t talk about this much but you can see semantic role labeling is kind of a general kind of information extraction. I’ll come back to that just with a little bit. And – okay, I’m gonna skip that.

So what are these semantic roles anyway? So here’s a traditional set of semantic roles that people have used for the last 40 or 50 years and I’ve used them intuitively before. So we have an agent, whoever did the volitional causing, an experiencer, a force, that would be like the wind that blew something down, the tornado, the hurricane.

A theme, so that the affected participant, the result – so some event happens, I’ll just skip some of these. And instrument – but instruments, somebody who benefits from the event and then when things get transferred from something to something else or when somebody perceives something from somewhere else, the source and the goal. So just some example sentences.

So here are regulation sized baseball diamond is the result of the building action. They built something – what they created was a diamond. Or here someone – it was an asking event and the result of the asking is a proposition, I’m sorry, the content of the asking is this proposition.

Here we have an instrument; a shocking device is used, in this case for stunning catfish and so on. Okay? So all things that you can imagine people asking questions about you can imagine if we’re doing information extraction of the kind you looked at a couple weeks ago that’d be nice to have this kind of relatively shallow level of semantics automatically pulled out of this stream.

Related to this and sort of more formally the idea that any individual semantic roles could appear in many different places and start with this easy one. So with the verb “give” it turns out that the theme can occur either directly after the verb “give” or later after the goal.

So I can have the theme before the goal or the goal before the theme and when I put the goal after the theme I have to use the word “to” in this case. So I have to say, “Doris gave the book to Carrie.” But I can say, “Doris gave Carrie the book.” I can’t say, “Doris gave the book Carrie” for example.

But either/or is okay. So when a verb has multiple possible orderings of this type we call this a diathesis alternation. So here’s one diathesis alternation and here for break a whole bunch of alternations.

So for break we can make the instrument be the subject, we can make the theme be the subject and this is a regular fact for a particular different verb have different and abilities to have their arguments appear in different places. So we can suffact that we can condition on the verb when we do our parse in a second.

So here’s the problem with this kind of semantic role that we just talked about. Is that no one has come up with really good definitions of roles and this very bad labeler agreement. If you ask a human to label a sentence so you could train some classifier humans disagree violently about what a role is.

And I’ve given you just one of the classic kind of examples here. So it turns out there’s two kinds of instruments called intermediary and enabling instruments. And it turns out that intermediary instruments are the kind that can be subjects of the verb but the enabling instruments can’t.

So I can say, “The gadget opened the jar.” Meaning that I opened the jar with the gadget. But I can’t say, “The fork ate the bananas.” So here’s a case where depends on whose labels that you use and whose list of semantic roles you might have two different instrument roles, you might have one. It might be hard to decide if one of these is really more of a force or more of an agent. It’s just a mess.

So this caused a huge problem in labeling and there were two solutions that led to the two large databases that we train our classifiers on now and they pick different solutions, PropBank and FrameNet are the two databases.

So what PropBank said was, “Well, it’s too hard to come up with semantic role labels so all we’re gonna do is assign them arbitrary numbers from zero to five.” And what FrameNet said is, “It’s too hard to define role names but if we restrict it to an individual semantic domain, a small domain, like say the domain of giving or something, then in that domain I could probably come up with some role names that are sufficient for that domain.”

And we're gonna see how to do each of these and then how to parse into each of those representations. So here's PropBank, is this big enough to see or no? Sort of? Okay.

So PropBank and FrameNet are two large labeled corpora, and this there – so we are gonna be doing supervised classification for this. So here's an example for PropBank. Here's a couple of example labeled sentences. So here's the verb, "agree" which has three arguments labeled, like I said, with just numbers, zero, one, and two, and there'll be a – this is called a frame file, a file that defines that zero is the agreer and one is the proposition and two is the other person agreeing with that proposition and here's a couple label examples.

"So the group agreed it wouldn't make an offer." So the group is our zero, some of agentive agreeing thing and here is the proposition that it agreed upon. And here there's two different arguments is John arg0, agreeing with Mary arg2 and so on.

And similar for fall. So sales, that's the thing that fell, fell to 251. So arg4, 251 is the endpoint of the falling. So we could fall from someplace to some other place by a certain amount. Each of those is a semantic argument. And so what PropBank is, is a large labeled corpus where all the arguments are labeled with zeros, ones, and twos. Yeah.

Student: Can you [inaudible] the – in all your examples we have 0 is always the [inaudible].

Instructor (Dan Jurafsky) :Good question. So just to repeat the question, is it the case that all instances of agree in a large labeled corpus would have the exact same definition of arg0, 1 and 2? And the answer is sort of.

So have you done word senses yet in class? Not so much? Okay. Anyway, words have different senses obviously. So bank the river and bank the money – bank the side of the river and bank the money – institution are separate senses. And the same is true for verbs so – I can't think of a second sentence for agree but so there are – whenever there's multiple senses for a verb there'll be multiple labels.

So there'll be agree 01, agree 02, let's say and agree 03. And for each of sense of agree it'll have the exact same mapping, 01 and 2. And so the PropBank corpus that was labeled it will distinguish the three senses of agree and then mark this is the arg0 of agree point 01. So yes, that's a good point.

And for some verbs they're very ambiguous. But they used a relatively broad notion of sense so it's not that – too many senses for each verb. Other questions?

Okay, anyway so why do we care about this? Why are we doing this? What's this – if it's just zeros, ones, and two what good is it? Well, the intuition is, again, just from the very beginning I would like to be able to know that if I see a bunch of sentences with increasing that if I wanted to know what increased, "What was the price of bananas?"

That in each case it's gonna be the arg1 even though the sentences may be different in all sorts of ways.

So if I have some question like, you know, "How much did shares of Google go up today?" Then I can find, you know, the price of Google as the arg1 automatically despite the fact that syntactically it may appear in various different places in the sentence. Is that clear? So PropBank labels are gonna helps us find the shallow semantic role of particular phrases. So far, so good?

Student: So it looks like they were trying to [inaudible] roles. But this is another, you know, [inaudible] arguments to [inaudible].

Instructor (Dan Jurafsky) :Yeah, they picked them up but think of it. They pick a much shallower abstraction. So instead of saying - I claim that this means that the person was agentive and had volitional action and all these complicated things, all I'm claiming is every time you see a sentence with "increase 01" I guarantee you that the arg0 will mean the same thing whatever it is.

So they don't have to define anything with respect to any other verb. So all they're saying is inside this sense of the verb agree I guarantee you that all the sentences that arg1, 0 mean the same thing.

So it's a pretty limited abstraction. It's barely abstracting over the surface form. So it's very, very shallow semantics. Yeah. And the goal is, again, you know, stepping slightly up from syntax from simple parsing toward meaning, toward information extraction. Other questions?

Okay, so what PropBank did – what did they actually label with this label set that I've now told you about? They took parse trees and they labeled constitutions of parse trees. So they took the Penn Treebank, which I'll talk about in a second, which – has Chris talked about that? Yeah.

So they took the Penn Treebank, so it's a million – approximately a million words, well, various pieces of the Penn Treebank. They took the Wall Street Journal portion of the Penn Treebank, so that's a million words, and they labeled every node in the Treebank, every time that a node was at some argument of a verb they labeled it with its role.

So in this case they would label this one whatever the correct instrument argument of hit is, let's say it's arg1 of hit or arg2 of hit would be arg2, I guess, they labeled this as arg2. So these went through all the trees in the entire treebank and hand labeled each one.

And I'm – oh, there it is. So they labeled Kristina as arg0, Scott as arg1 with the baseball bat as arg2 and yesterday as argtemp. So the – they distinguished between – I think I might have a slide on this but just to preview the core arguments, the numbered ones 0 to 5 and then a few kind of semantic-y ones that look a little bit more like the kind of named entity tag you guys looked at alright, temporals, locatives, numbers, other things that might play a role in the semantics of the sentence.

So semantic role labeling sort of shades into named entity tagging. You guys have had named entity tagging already? Yeah. Yes? Yeah, have they had named entity tagging? Have they had named entity tagging? Yes? Okay, good. Okay, I didn't get enough nods there, you guys.

And so I want to point out that there's another way to think of this, and I'll come back to this later, which is instead of labeling the parse tree I could just think of this, again, just like with named entity tagging as labeling chunks of words with labels.

And both of them are possible algorithms and most people turn out to use the parsing one, it seems to work a little better. But I think in the long run we may move to this kind of more flat representation.

Okay, so this is just another example. Skip that. Okay, so this is PropBank. I'm gonna ignore all this stuff. Never mind the details of representation. I just – the slide is there in case you need to go back and look at it if you're doing [inaudible] projects with this.

But here's the details. So the frame files are those files I showed you with one entry for each sense of each verb. English has about 10,000 verbs but a lot of them don't occur in the Wall Street Journal.

And recently – so Martha Palmer is the researcher at the University of Colorado who built the PropBank. She's also built the Chinese PropBank and then Adam Meyers at NYU has built a NomBank. So PropBank is just verbs, the Chinese PropBank is just Chinese verbs and NomBank is just nouns.

So instead of saying, "John gave a book to Mary." We can say, "Mary's gift of a book to John." So gift is a noun but it has the same – we want to be able to extract the same kind of arguments from nouns as we do from verbs. If I want to know who gave what to whom I might have expressed the giving as a noun or a verb and either way I want to know that John got it, let's say. Is that clear? So it could be nouns or it could be verbs.

So it just happens that PropBank started with verbs and then NomBank moved on to nouns. It could have been the other way around but that's just the way it went. Questions so far? I'm gonna skip that.

So here's the problem with PropBank which is okay, PropBank doesn't have any nouns, NomBank has the nouns but in both cases they don't tell you how to link across different words. So PropBank just guarantees that if there's a verb increase that every instance of increase with the same sense will – arg0 will mean the same thing.

But I want to know sometimes I might use the verb rise or the noun rise instead of increase. So if I know that the price of bananas – I want to know if the price of bananas increased or how much it increased by but the sentence that I'm grabbing the – trying to parse has the information says that the price of bananas rose, PropBank can't help me tell me that the thing that rose, the arg2 of rose is the same as the arg2 of increased.

Because the actual numbering of the arguments are only specific to an individual verb. So we'd like to be able to generalize across particular predicates. So FrameNet is designed to do that. So what FrameNet is, is a series of frames, so there's the hit target frame where I say inside this frame, and a frame is sort of a semantic grouping of things, inside this frame I'm gonna define some names of roles and those are gonna be good for any verb in this frame.

So any verb of hitting or shooting or picking off, and for any of those I'm gonna define names like agent, target and instrument and I guarantee you that my labels for any of those verbs – the semantics of the word instrument will be the same.

So if I want to detect who got hit and the word target is used in my label data I'm guaranteed that for all the verbs of hitting it'll be the word "target" in the labeled data. So I can generalize from one verb to another. And they use various terminology like frame elements and – well, I won't go into that but you get the idea. Semantic roles are called frame elements.

So I'm gonna walk you in a particular example. So here's a particular abstract frame called the "change position on a scale" frame. And so here are the roles used for this frame. So you can see they're very different than things like agent-patient theme instrument.

So this is, you know, "The stock fell by $10.00." So there's the attribute, that's the property it possesses, the distance, how much it fell by, the final state where it ended up at, the initial state, the initial value and so on and the range.

So let me give you some sentences with some of these values filled in. So if I say, "Oil rose in price by 2 percent." Oil is the item that changed, price is the attribute that changes, it could have risen instead by – in volume or density or something, but it rose in price, 2 percent is the difference between the initial and final states and so on.

And there's lot of verbs, lots of nouns and even some adjectives that are in this frame. So for any of these verbs or nouns I'm guaranteed that if I want to know what the change – you know, whether Google stock – exactly how much it rose by in the article it might have – that I'm searching might have used the verb "plummet." I'm guaranteed that the whatever it is, the difference semantic role in some sentence labeled with "plummet" is gonna tell me the answer. So far, so good? Questions? Totally clear?

Okay, so FrameNet has a different problem. Unfortunately neither of them is perfect. The problem with FrameNet is instead of – in PropBank they just label the whole corpus, so we have a million words labeled which means that the frequencies with which things are labeled are proportional to the actual frequency in at least the Wall Street Journal.

With FrameNet on the other hand, they just randomly selected a sentence as a good example for each verb, ten sentences has good examples for each verb which means they're not randomly selected and they're biased in various ways. And more important it means that the entire sentence wasn't labeled.

So if I want to label in FrameNet an example of – I didn't put any examples with multiple verbs. Lots of real sentences have multiple verbs. In FrameNet only one of them would be labeled in a particular sentence.

In PropBank every single verb in the whole Penn Treebank was labeled. So that makes it hard to build classifiers because I don't have examples that are randomly selected which means their distribution is the real world and I don't have the whole sentence.

Furthermore in PropBank the treebank was labeled so I know the exact correct parse. In FrameNet, I don't have correct parses because they labeled random sentences from the British National Corpus, which isn't parsed. Or actually is it parsed now but not hand parsed. And FrameNet's still ongoing so that's good and bad. It's good in that it's getting larger and larger every year but it's also changing all the time.

Okay, and I just wanted to mention very briefly some history, because you should know about some history. So this idea of semantic roles came out of Phil Murrow's work in 1968. Question?

Student:[Inaudible].

Instructor (Dan Jurafsky) :Yes.

Student:[Inaudible]. So what exactly did the [inaudible] do?

Instructor (Dan Jurafsky) :Good. So the question was since they're complimentary, PropBank and FrameNet, couldn't you train some classifier on PropBank, get the PropBank roles, map them to the FrameNet roles, and then have some kind of more semantics?

And yes, something like that is a really good idea and people have been trying to build a FrameNet to PropBank and PropBank to FrameNet mappings. It's not trivial. It turns out it's messy. Part of the problem is the verb senses are different.

So in FrameNet you have to decide which frame you're in because – so the same verb might – so the same verb in FrameNet might appear in different frames. So I don't know, rise might be in the – let's see, let's get a good example.

Student:[Inaudible].

Instructor (Dan Jurafsky) :They are but they might have picked different ways to distinguish the senses. So in PropBank increase 01, increase 02, increase 03 might not map into FrameNet exactly but increase 01 goes to one frame, increase 02 goes to another, increase 03 goes to a third.

So that's the problem. But it's clearly the right thing to do and people – and that's something that's actually – I would say is an ongoing research project. In fact, some students in our lab have been interested in this exact topic.

So the clear idea is, you know, do some kind of learning where you've got two separate databases. You've got really two independent sources of evidence for some kind of shallow semantic roles. It would be stupid if we couldn't figure out a way to combine them. Yeah.

Okay, and I guess I just want to say, without giving you a whole history – I'm not gonna give you the whole history, but a key fact that you should know is that Simons at the University of Texas at Austin really in 1968 he started. In 1968, he built code for doing semantic roles labeling by doing parsing first and then taking the output of the parser and saying, "Look, I've got my parse tree. Oh, that noun phrase before the verb, that's probably the subject. And the subject, that's probably the agent if it's verb x, y and z." And so on.

So he did this in 1968, he used it for a question answerer and the question answerer he built in 1968 actually turns out to be almost exactly isomorphic to a modern question answerer that [inaudible] built this year.

So he – a lot of the modern work on parsing – semantic parsing and question answering was really pre-figured by the '60s. But they didn't have large databases; they didn't have a lot of code. So, like, he gave all his code in, like, two pages of lists in his first paper. And it certainly didn't have [inaudible] ability to generalize from large datasets but it was all there. So everything we've been doing since then really is just getting his argument to work in large scale I would say.

Okay, so FrameNet, this is Chris's slide, it's a little tongue-in-cheek. So FrameNet basically – where in PropBank the goal was just go through every sentence in the Wall Street Journal labeling every verb.

In FrameNet, the goal is come up with a new frame for that frame to find a whole bunch of verbs and nouns, find sentences, and then annotate them as opposed to going through a particular corpus word by word. So it's a, again, different corpus. Okay, and here's a pointer to FrameNet.

So one way to think about the difference between FrameNet and PropBank is with buy and sell. So in PropBank if Chuck bought a car from Jerry or Jerry sold a car – oh, let's do it in FrameNet first. If Chuck bought a car from Jerry then probably Jerry sold the car to Chuck. It's not absolutely 100 percent – you could imagine it not being true but it's very likely true.

In FrameNet, you can capture that generalization because Chuck is the buyer of buying and he's the buyer over here in the selling sentence as well. In PropBank Chuck is an arg0 here and an arg1, arg2 here. So you can't capture that. So PropBank is sort of closer to the surface form. You could think of it that way. And FrameNet's sort of one slight level more semantic.

And this is, I guess, partly explains the difficulty in your task which is if I had a PropBank labeling to map to a FrameNet labeling I have to know that the arg2 of selling turns out to be the buyer but the arg2 of buying turns out to be the seller and so I have to learn that.

Student:[Inaudible].

Instructor (Dan Jurafsky) :Yeah, good question. So the question was – I'm assuming I should repeat the questions for the taped audience. So the question was suppose you didn't have FrameNet, could you just take these two PropBank sentences and notice that the subject of buying often was the same word as the object of selling?

So you could use some kind of distribution over possible fillers of words and notice these two distributions are similar. And could you automatically learn fragment [inaudible]? And that's actually a great tasks. So people have actually proposed sort of inducing both PropBank and FrameNet either full automatically or semi-automatically by doing just that kind of thing.

So I'd say yeah, but unsolved perfectly good final project for this class. I think Chris would love that. If you haven't picked a final project yet – I guess you must have all picked them but just in case you haven't that's totally a good final project, very publishable.

Okay, I'll skip that. So just a little bit thinking about how – to think about this with respect to the information extraction and stuff you've already seen and in both cases we can think of these as filling frames and slots.

So information extraction is I've got a bunch of slots, like, I don't know, some – I want to know which companies acquired other companies this year. So I'd be acquiring company and the acquired company and the price they paid, those are my frames. And the fillers are the names of the companies that got acquired and so on.

And the same is true for semantic role labeling. We have, you know, our roles are the agent and the patient instrument or the amount that the thing went up and how much – and where it got to, and so on. But you can think of semantic role labeling as being much broader coverage version because it replies to every verb. Whereas information extraction I've got to tell you the very specific domain I want to extract the information from.

They're both pretty shallow whereas IE is generally application specific. We want to extract some – fill some frames and slots in because we've got something we want to do with it. Semantic role labeling is sort of like parsing, it's this general tool we throw at a sentence and hope that these roles will be useful for something.

Okay, so now I want to define the actual algorithm. I've spend the first little bit talking about the representation and now our job is I'll give you a sentence, you give me that representation as the output.

And again, it's gonna be supervised classification and that's how people do this now. Although people started to think about how to do this in an unsupervised way as you were suggesting. And again, a topic where both Chris and I are actually quite interested in.

Okay, so I'll tell you about the task, how to evaluate it and how we build it. So people have generally talked about three tasks. So identification is just looking at a parse tree every possible node in the parse tree tell me which of those nodes are arguments, which of those noses are not arguments without even giving them a label. Just identify binary classification, yes, I'm an argument, no, I'm not an argument.

So in a very large parse tree you can imagine there's lots of places – most things are not arguments and only two or three things around each verb are the arguments and all their sub pieces are not arguments and so on.

So that's – identification is hard. Classification is to – if I give you the node and the parse tree and therefore a string of words I tell you that's a constituent you can [inaudible]one of them labeled which for a particular verb might be 5, you know, arg0 through 5 plus the things like temporal and locative and those kind of things.

Core argument labeling is slightly easier because it's just giving you the label 0 through 5 in PropBank and not having to worry about temporal and locative and those other kind of things. So it's really mostly subjects and – the subject and the direct object is most of what core arguments are.

Okay, so how do I evaluate this? So how do I know? Suppose I built this – I haven't told you algorithm yet but I'm gonna make you figure out the algorithm and you can figure it out.

Suppose I had this semantic role labeler or how do I know if I've done a good job? So up there we have a correct hand built answer. "The queen broke the window yesterday." So arg0, arg1 and then argtemp and let's say our system produces this guess, "The queen broke the window yesterday" where we've not quite got the right constituency right for the arg1 and we've mislabeled the argtemp as an arg lock.

So here's the two ones we got wrong. And so just like any other task where we're labeling a bunch of things we can give you precision recall and F-measure. So we can tell you of the things that we returned how many were correct, these two. Of the things how many were wrong, these two.

Of the correct things how many did we find, we found two of them, and so on. So we can define position recall like we can with any task where we're trying to pull something out. And so – okay, so what's wrong – sorry, I meant to ask you this before I went to the next slide. What's wrong with this evaluation?

Student:[Inaudible].

Instructor (Dan Jurafsky) :Okay, so there's no partial credit. So the window and window, we ought to get some partial credit for window. Give me a bigger problem. Step back from this task. Suppose you're a venture capitalist. Why would you be unhappy if I gave you, my funder, these numbers?

Student: Because it's a very easy example.

Instructor (Dan Jurafsky) :Okay, it's easy example. Okay, so I should give you a harder example. Okay, so I gave you these numbers on a really hard example? Why should you still be suspicious?

Student:[Inaudible].

Instructor (Dan Jurafsky) :Good.

Student:[Inaudible].

Instructor (Dan Jurafsky) :Okay, good. So one more complaint you could say. So one, no partial credit, two, this is a semantically important distinction, like, this is – the whole point is to get the semantics right and it gets the semantics wrong. Like I can believe a parser can find subjects and objects but – so I would imagine that's pretty good. Well, it's kind of unnerving that it can't get the semantics right on something that should be really easy.

Okay, go even higher level. Again, think like a venture capitalist. If I asked you to build something and you tell me, "See, I can make the human labels on this test set." And why is that not good enough? So okay, maybe I just asked this question in too vague a way. So there's two ways to evaluate any NLP system and they're often called intrinsic and extrinsic evaluation.

So intrinsic evaluation means I'm labeling some data with some label and my evaluation is did I match the human on how they labeled the training set from which I got these labels. Extrinsic evaluation says I'm gonna put this labeling system into something that actually does something useful and see if it makes it better.

So let's say I got my beautiful semantic parser and let's say I make it, I think 5 percent better, meaning that it matches the human labels 5 percent better. That's not very useful for anything unless I show that it actually improves; let's say question answering or machine translation.

So this is an example of intrinsic evaluation. You should always be skeptical of intrinsic evaluations because intrinsic evaluations are fine for, you know, testing your system to make sure that you're not making a mistake or that you show an improvement but nobody but people in your sub-sub-sub field care about your results and your intrinsic evaluation.

So whenever you build any tasks like this you must think about extrinsic evaluation. What task is this gonna make better? And if it's not making some task better why am I working on it?

Student: Can you [inaudible] that, like, one of the reasons we had like, you know, this has a specific number of different frames and you might say well, [inaudible] says and when somebody's [inaudible].

Instructor (Dan Jurafsky) :Exactly. So yeah, why could this be bad? So it could be that the actual data for some real task is all proteins, lets say it's about some biology data. Well, this is useless on that. So you don't really care what the numbers look like on this data. What you care about is the actual data you're gonna see and it might be from different domain, might have different verbs, even if it's not proteins, even if it's newswire text.

It could be that – well, this data's from the Penn Treebanks from 1991. Well, the new data all the nouns are different because that was 17 years ago and none of the names of the Presidents are the same or whatever.

So the company names have changed, the airlines have all changed, right? The major airlines being discussed in 1991 were TWA, which isn't around anymore, and a bunch of other ones that aren't around anymore. So it's probably gonna work really terribly on any modern data and so any task that uses that you won't notice that if what you're matching is the human labels.

So people have often called this the natural linguist tasks. So some linguistic grad student labeled the Penn Treebank and if we're just gonna measure how well we can match those labels that's – that can't be as good as an extrinsic evaluation where we actually build something with this. So I just want to point that out.

Okay, all right. So here's how most semantic role labeling systems work, relatively simple. Three steps, what – this is Kristina's slide. I would call it feature extraction. So we see a raw sentence, we extract a bunch of features; we parse it and extract a bunch of features.

Local scoring we decide for every particular constituent just by itself what we think it's likely to be. And then joint scoring we do some global optimization where we say, "Well, we probably don't have two different things that are both arg0's so we can do various things to do a global optimization over these local scores." Okay, and that was just very high level but I'll give you examples of all this.

So the first thing is almost all semantic role labelers just walk the tree. The run a parser, take the output of the parser, walk the nodes in the tree and just run a classifier either – and I'll just talk about that in a second, run a classifier to each node.

So I run a classifier in this node, the PRP node and say – often it's two stages, I say is this – forget the two stages, let's suppose it's one stage. I just run in this case here's the verb, broke, so I run a classifier that says, "Oh, I know break in my training data has 5 arguments so I'm gonna build a classifier that knows the verb and has a bunch of other features and its job is to assign a label to PRP and to NP and to BVD and to VP" and so on.

And for each of these it's gonna assign a label like arg0 or null, none, I guess none, and so on. This one will be arg1 and this one will be nothing and so on.

So I'm gonna walk through the – create just the final label to every node in the tree. So the alternative, I mentioned this earlier, is instead I do some kind of linear walking through the strings like a named entity tagger and I make a – yes, I'm at the start of an NP decision. I'm sorry, yes, I'm at the start of an arg0 decision here and I make a – I'm at the end of an arg0 decision here and so on.

I linearize these things like you saw with the entity tagging. But this seems to work better. The reason it works better – oh, no. Wait. So first I want to say – well the reason it works better is that if you actually look at PropBank and FrameNet it turns out that the things people labeled as the semantic roles turn out to be parse constituents, turn out to be pieces – they turn out to be conveniently be nodes in parse trees 90 to 100 percent of the time depending on how you look and how you count.

Even if you look in Charniak and Collins or [inaudible] that Chris will go over – I think next lecture actually, at least he'll still go over Collins, I think. And even if you use automatic parses as opposed to human goal parses it's still a case that most things are argument. And the reason is that in PropBank people actually labeled these by looking at the parse tree and marking their labels on a node of the parse tree so you're guaranteed that what you've got is a parse constituent.

So it makes sense to do parsing first as opposed to say entity tagging where the name entity is the times and locations might have been labeled in a string of words and not in a tree. And so you're not guaranteed they're gonna match somebody's idea of a tree.

And even FrameNet the people who labeled these things – even though if they didn't have parse trees they were trained in syntax, they were trained in parsing first, then they labeled things. So there's this bias towards parse trees. Yeah.

Student:[Inaudible].

Instructor (Dan Jurafsky) :So are you asking whether semantic – so the preposition phrase attachment problem is a classic problem in parsing. So ask the question again. Are you saying would semantic role labeling help or is it subject to the same problems?

Student:[Inaudible].

Instructor (Dan Jurafsky) :You're saying that if you get the – if you attach incorrectly the preposition in the parse tree will you also therefore get the semantic label on? Yes, yes. So parse errors are the major cause of semantic role labeling errors and the reason is that, again, people – the natural linguist task we're trying to match the labels in some database and the labels were put there on the trees.

And so if your parse tree is wrong you're gonna get the wrong labels. So, I don't know, if you have a parse attaching to a noun phrase instead of a verb phrase and it's supposed to be that these two were separate arguments then you're gonna incorrectly think they're the same argument somehow.

So that's, I think, one of the main issues – preposition attachment and relative clause attachment and various noun phrase often parsing problems are what cause errors in this, yeah. Other questions when we stopped?

Okay, so here's another – I just gave you another slide. This is from Sameer Pradhan's slide. Here's another way to very high level just more of a processing way to think about all algorithms I know of for semantic role labeling.

So you parse the sentence and then you just walk through every predicate, meaning verb or and later stages now nouns. And for every predicate you look at every node in the parse tree, extract a future vector for that node, classify the node and then do some second, you know, pass. So I kind of said this earlier but this sort of makes it more explicit where the algorithm is. I guess that's what we're gonna do.

Okay, so now I'm gonna build a classifier. It's gonna extract a bunch of features and classify the nodes. All I have to do is tell you what the features are because the rest of it's just machine learning, right? At the training set I've got a bunch of, you know, parses and in each node I tell you what the right answer is, I give you a bunch of features, you train a classifier, I give you a new sentence, you parse it, you extract a bunch of features and you classify each node.

Is the algorithm completely clear? Totally intuitive? So all I have to tell you is the features and we're done, right? Okay, so this – a lot of the modern work on semantic role labeling came from the dissertation of Dan Gildea, one of my students, who was at Berkeley. And so these are the – they're often used as baseline features for everybody for their classifiers now. Some I'm gonna talk through Dan's features.

So the first feature is the verb itself, so we have talk. Another feature is – and this is an important feature, this comes up over and over again now sort of after Dan's work in all modern – anything where you're using the parse tree as the first step you tend to use this feature called the path feature.

And so the path is a flattening – it's a way to flatten out a parse tree. So I want the path from the constituent to the predicate. So let's say my constituent is he. I want to know how to get from "he" to "talked." So I go up to an NP, up to an S, down to a VP, down to a VBD and that's the flattened path.

So it's a flat way to represent a path and so this kind of notation or similar variance of that notation and the idea of throwing in this kind of string or variance of the string is a path came at a down – Dan sees it – that's probably the single most important thing I can tell you about the features. The other features aren't so important for the rest of – you remember the rest of your life. But anytime you need a parse throw in this flattening of parse feature turned out to the efficient way to do things.

So other things – the phrase type, so if we're trying to label an NP knowing that it's an NP is obviously helpful. Whether I'm before or after the verb because, you know, arg0's often happen before the verb; arg1's often happen after the verb, whether I'm passive or active, that's a useful thing.

In passive sentences it's often the arg1, the theme that comes before so, "The table was built," table is the theme or result. The word itself so if I'm – and I think Chris will talk about headwords. Has he talked about headwords yet? I think not until Wednesday.

So when you get to lexicalized parsing every phrase has a headword. So the headword of "about 20 minutes" is minutes. And the headword is semantically the most useful thing. It gives you a hint this is likely to be a temporal phrase whereas the "about" doesn't tell you that much.

So we throw in the headword and we throw in some categorization. So in this case it's we know that the verb has – is a verb that in this case verb phrase the verb is followed by – has Chris talked about some categorization?

No, I think that's also next time. So that's a kind of lexicalized parsing feature. I'll wait for him to define next time but it basically says certain verbs expect to see prepositional objects so talk for 20 minutes. Other ones like "bake" expect a noun phrase, like, "I baked a cake."

And the kind of arguments that a verb expects are called the verbs of categorization and we can talk about it also as what are the list of things in the verb phrase and that's a useful thing for telling us a fact about whether – what kind of semantic arguments I've got.

And here's a bunch of other features that most of which are used now in all classifiers. People just throw in 20 or 30 features and maybe I'll give little definitions of some of them. But instead of using the full path we could just use part of the path so maybe just a path up or just the path down.

We could run a named entity tagger and see if the named entity tagger things we've got a location and we have double evidence we've got a location. We can throw in, you know, words nearby. We can throw in – and Chris will talk about this soon about – have you had [inaudible] labeling and parsing yet?

No? Okay, fine. So apparently all these various things that we can do in parsing we can throw in all the neighboring stuff in the tree basically as features.

So we can throw in – if we know what verb it is we can throw in the class of other verbs in that class. So maybe in our training data we didn't see this exact verb but we did see some other verb that's like it. So now we can generalize more generally and we can do that by using WordNet which you'll get to which is – have you had WordNet? No. Or you can do distributional clustering of the type we were talking about earlier and just figure out these words are similar automatically. You know, various other words.

The preposition identity, pieces of the path, a named entity tagger telling us this is a time event and so on, this and that and the other thing, various pieces, various ordering. They're like lists of words, lists of temporal cue words that helps us identify temporal things. Okay, this is what people do.

So, two ways people think about semantic role labeling. Method 1 is we first run quick efficient filter that just throws out parts of the nodes so that just for efficiency we don't have to run a classifier on every single node in the tree because parsing takes a long time and then if we have to look at every node in the large parse tree most of them are useless.

So we run some quick heuristics to throw out most of the nodes leaving just the one that are – have some potential of being a constituent. And then we run our slow SVM or some other kind of classifier just on those individual node and then we do – sorry. So first we have a filter, then we do a binary classification along those to throw out some more, and then we actually label each of the nodes that are left. Okay?

We can also do the whole thing in one step by building a classifier that just says give me the label given a whole bunch of features of the parse tree, so instead of doing this three step process. And people have done both. And, again, it depends on efficiency and how slow the algorithm is for looking at each node.

And some cool recent things people have done, so Kristina has a really cool – Kristina and Chris have a really cool paper which I think is the – I think they have the best published results on semantic role labeling by jointly training a labeler to label the entire tree at the same time. So labeling all the nodes jointly conditioned on the whole tree. Because if we do it independently, if we just ask, "Am I an arg0? Is he an arg1? Is he an argtemp?" That's a different classification than if we optimized the entire tree at the same time.

Okay, so how to think about this? So here’s ways you can do that. So, how to think about this? I think it’s better if I give you another – yes, do this slide first.

So what kind of joint global constraints might I use for improving my classification of an individual node? Well, one is they don’t tend to overlap. So if I’ve got a big huge noun phrase and inside it’s a smaller noun phrase it’s unlikely that they’re both arguments because that’s just the way it works. A verb’s arguments tend to be separate strings of words.

So we can enforce that in various ways. We can make that either a hard constraint or a soft constraint. So these are all ways of sticking [inaudible] a hard constraint into the search. So we can say, “Look through all possible nodes in the tree, assign each node a probably and now find me the best covering of the words that give me the highest probability of the sequence of nodes.”

Or I can do that – instead of doing that as a greedy search I can do an exact search or I can use other algorithms like [inaudible] programming for the algorithm.

Other heuristics that you can throw in are that you don’t get repeated core arguments. You don’t tend to get arg0, arg0, arg0, arg0. You don’t tend to get a bunch of agents in a row, you usually only get one of each. And, of course, the predicate doesn’t tend to occur in the – a phrase doesn’t tend to jump over a predicate. They tend to be consecutive and so on.

Okay, and what a lot of people have done is throw in like a language model. So you can say well, it’s likely to be the case for certain verbs that arg0 is followed by arg1 and that’s like 80 percent of the time.

So we could actually think of that as a language model. We’re just predicting – give me the likelihood for the joint probability of the whole sequence arg0, arg1, argtemp given the verb, let’s say. And we can throw that in as a feature with the probability of this entire sequence; train that on our training data. So that’s what I forget whether which of these – I think Sameer did that.

Or what Kristina did is build a whole joint model with – based on the conditional random field. It took into account all of the possible sequences of arguments and that’s what I think is the best-published numbers on this task. So if you’re interested in improving semantic role labeling 2-10 of ’05 is the model to start with.

Student:[Inaudible].

Instructor (Dan Jurafsky) :Yes, it’s much – yes, it’s hugely slow, hugely slower.

Student:[Inaudible].

Instructor (Dan Jurafsky) :Vastly slower, yeah, yeah, yes, so, yeah. And speed has been a problem in all these things. I mean, parsing is in general – first of all you've got to run the parser first and parsing is one second per parse which is fine for one sentence but if you've got to retrain your data and you've got to parse the whole train set, it's just a pain in the butt. So, yeah. And – yeah, very slow. So this has been a problem. And also people have done clever things with tree kernels and tree kernels CRF's and people have tried pretty much everything.

What I didn't say is that there's a bake-off every year. Do you know what a bake-off is? We talked about bake-offs. So a bake-off is when you – when everybody who lives on a street they all bake a cake and then you have a little contest and the winner gets First Prize for their best cake.

So in NLP there are bake-offs for everything every year. So it's kind of a joke name. But there's bake-offs for machine translator every year, there's bake-offs for semantic role labeling every year, there's bake-offs for speech recognition every year.

And so there's been lots of bake-offs on semantic role labeling so lots of people pitting their systems against each other. So we have a very good idea of what works best because we see 10 systems every year trying everything.

So I think is Kristina's slide showing that if you do – just looking at the core arguments if you do joint labeling, a CRF where you look at all the labels at the same time rather than individually optimizing first this constituent, then that one, that you get improvements, let's see, joint is the purple one.

So you get improvements over either doing them locally with their better features or the best published numbers before them. So this is, you know, a standard AI thing that everyone knows is that if you can solve a problem jointly you can always do better if you can figure out a way to do it. So they figured out a way to do it where you can look at all the constraints at the same time rather than breaking it up and [inaudible] independence.

Okay, so here's a little summary. So most modern systems just use a standard set of features. The features haven't really changed in the last, I'd say 5 years. And they throw in lots of them so you parse; you extract a ton of features. It doesn't seem to matter what learning method you use, although there's an advantage to being able to figure out how to solve the problem jointly. So you may make that easier but you can do that in other formalisms if you want.

And this is all information that we know from the bake-off so we've seen systems against each other. So you're gonna learn about the Collins and Charniak parser are the two best – the two best available, best performing parsers in the word.

Probably the number three right after those is the Stanford parser one that we have which we often use because we control the source code and it's much cleaner. But in terms of actual performance those two are slightly better and it does seem to matter the quality of your parse. And then various ways of combining either information from throughout multiple systems of the problem, like build three semantic role labelers and have them vote. That's very common.

Or reranking, you know, run a really fast classifier and have it output five possible semantic role labelings and have your really CRF expensive slow one just look at those five and pick among the five and rescore them. That's a very common way to use very – if you've got a very expensive algorithm you run it as a second pass as a reranking.

Here's a little error analysis. So this is from Kristina's results on – from CoNLL. That's the Computational Natural Language Learning conference that hosts the bake-off for semantic role labeling.

And so you can see that arg0 is really easy, arg1 is not bad but by the time you get to arg2 and arg3 things are not going so well because your numbers that are not in the 90's, you know, if you're having to guess 20 percent of the time on the semantics of something and this is after a parser and this is gonna be put into some other system it seems pretty sucky.

And definitely things like manner or location are having huge problems or adverbial. And even temp, you know, ought to be a lot better than 78. Temporal phrases are pretty – should be pretty easy to spot.

So I would say there's lot so of room for improvement in the field. But again, these numbers are all, you know, "How well can I match a human label?" So they're hard numbers with no partial credit, you know, the humans might be wrong. We don't know if the things we're good at are the things that matter.

I mean, I suspect finding arg0 just doesn't matter very much. I mean, the fact is that semantic role labeling – well, we'll come back to that. So we don't know which things to improve and what – until we really look at the application we stick this in to see if it really helps the application.

Okay, so what have I said so far? Okay, so semantic role labeling is pretty successful, not perfect yet. By no means is it a commercial quality task I think, at getting human labels in these two tasks that people have been using for years. And we just don't know yet if they're good for anything. So there's a number of papers where they've shown a 1 percent improvement in performance of some algorithm by throwing in semantic role labels first.

And you never know with 1 percent improvement if that's just because you looked at the problem harder. And if you put in the features the semantic role labelers was looking at instead directly into some other tasks you just – those features were helpful features and then you need to run the slow semantic role labeler just throw in those features into some other tasks.

So we don’t know. I’m – I’ve been working on this task for a long time and I guess I’m pessimistic that this is really gonna be really great for anything. But it might be perfectly okay for anything.

I mean, I think I believe – I think Chris and I might disagree on this. I think I’m not convinced that parsing has been shown to be good for anything yet even though we all do a lot of parsing and we all work very hard on improving our parsers. I think there’s no algorithm that parsing has unambiguously helped. But I think it’s clear that parsing will help for machine translation soon and maybe semantic role labeling, too. Okay?

Student: What about all the question and answering [inaudible]?

Instructor (Dan Jurafsky) :Say that just a little louder.

Student: What about the question and answering [inaudible]?

Instructor (Dan Jurafsky) :No, search doesn’t – parsing has not – never helped for search, probably never will. For question answering – well, it never will that’s a dangerous thing to say, but certainly hasn’t helped until now. People have been trying for 40 years.

Parsing for question answering, yeah, I mean, question answering’s a funny one because it’s not really a commercial application yet. And so yes, in human evaluations of factoid question answers, parsing alpha semantic role labeling have helped. So it’s possible that – I mean, question answering is the obvious place to look. So everything that we’re doing now we’re trying to show that it helps question answering.

And I think the answer is yes, it seems to help question answering. The problem with – that unlike MT and unlike search and unlike, say, spam classification question answering isn’t yet something where everybody’s using it out in the world and we can show that we can improve a commercial product by 1 percent. So the things that we’re improving are laboratory system. So – but yes, I thin that’s the place to look, question answering.

Student:[Inaudible].

Instructor (Dan Jurafsky) :Summarization – so yeah, good question. There’s multiple kinds of summarization. So most – have you done summarization yet in class? No, okay. So most summarization these days is extractive summarization. There’s two kinds of summarization, extractive and abstractive.

Extractive is, “Pull me out the sentences that are related – take a document, pull me out four sentences from the document, maybe clipping and pasting a little bit and create me a summary.” Abstractive is, “Write a summary that has the meaning that’s in these that may not use the words.”

So most modern summarizers are extractive because it’s much easier and if you’re trying to clip off parts of the – since you’re trying to – if your requirement for your extractive summary is that it be a certain length, like it’s got to be a snippet, let’s say, Google’s displaying a snippet. You want that summary to be very short, then parsing may help with that.

And semantic role labeling may also help, too. I hope. But there’s no evidence for that but parsing may help because it helps you know which things to clip off that are constituents. So yeah, I think extractive summarization will be helped by parsing and maybe has been helped by parsing already. Other questions? Any other billing ideas for applications we could apply semantic role labeling for?

Student:[Inaudible].

Instructor (Dan Jurafsky) :Oh, you can – you mean dialogue systems? Yeah, okay that’s a good question. So information extraction, which you’ve seen, semantic role labeling which you’ve just seen, and dialogue systems are extraordinarily similar.

So in all cases sort of all successful cases of meaning right now in Natural Language Processing use frames and slots or you prespecify some list of things you’re looking for and your goal is just to fill in the thing that fills that frame.

So, “Find me the arg0 and arg1 in this sentence.” Or, I don’t know, information extraction, “Find me the acquiring company in this news article.” Or your job is to parse obituaries, “Find me the name of the guy who died and when he died.” So you give a list of information you want and then you, you know, “Find me the price of all computer monitors from this website.” So you know what you want. You want the name of the monitor and its price. So – and your job is just to build systems that will fill in that slot.

So that’s true for information extraction. You’ve seen it’s true for semantic role labeling. It turns out to be how and conversational agents work. So let’s say you’re building a system that was built for – by nuance for United Airlines that you can call United Airlines and book a flight.

Again, there’s only seven things you could be giving it. The flight number, the date, the time, the airline. So we know in advance what the seven classes we’re trying to learn are and now our job is to build a classifier that will just fill in that information.

So yes, the same technologies that people think about for semantic role labeling they use very much for dialogue. Dialogue it gets more interesting – there’s like really cool problistic algorithms for dialogue for markup decision processes where you want to not just extract from one sentence to pieces but you want to figure out what you should say back to the user, you know, or to get the piece they didn’t tell you.

And so you sort of want to optimally ask them the perfect question just to fill in what you don’t know. But, yeah, those three are in fact all very similar algorithms and that’s where sort of shallow meaning extraction is what we’re all trying to get to work right now as opposed to very deep meaning extraction which is also interesting but just, you know, further away.

Other questions or suggestions? Okay, I'll stop there. We'll end a few minutes early. If you have more questions you can come up and ask me.

[End of Audio]

Duration: 64 minutes