NaturalLanguageProcessing-Lecture01

Instructor (Christopher Manning):

Hi, everyone. Welcome to the first class of Stanford's cs224n, which is an intensive introduction to natural language processing concentrating primarily, but not exclusively on using probabilistic methods for doing natural language processing.

So let me just say a teeny bit about the structure of the course, and some of the administration. If you want more information on any of these things, the main thing to do is go to the website, which is cs224n.Stanford.edu.

So I'm the instructor, Christopher Manning, and there are two TAs this quarter, Paul Boundstark and Dejuan Chang. The lectures right now are Monday/Wednesday 11:00 to 12:15 being broadcast live. Most weeks there'll be a section on Fridays from 11:00 to 12:15. So for the handouts for today there's course syllabus, there's the first lecture, and then most importantly I'm handing out the first assignment already today, and I'll say a bit more about that as the class proceeds. But this would be a good time to over the weekend look at the first assignment and check that you know how to do all the kinds of things that you need to be able to do to get productively working on the assignment next week in particular. So for this class there are three programming assignments that respecify to you, and then there's a final project, and most of the grade is that work. In addition to that, there are going to be just a few percent on weekly quizzes just to check that people are vaguely keeping up with all the other topics. But we're seeing this as primarily a project based learning kind of class where a lot of the learning goes on in doing these projects.

And so for the first three projects, all three of those projects, have a lot of support code for them to make it easy for you to do interesting things in the two weeks you get to do them in, and all of that support code is written in Java. In particular, it's Java 1.5 for generics and stuff like that. So basically, a requirement for this class is that you can handle doing the Java programming. And so hopefully you've either seen some Java before, or it won't be too difficult to kind of catch up to speed. Java's obviously not that different from other languages like C++ or Python the way most things work. And you know we're not – it's not that we're heavily using system's libraries beyond a few basic things such as collections for razor lists.

Most other information, see the course webpage. The little note on the bottom was I surveyed people where they actually want paper handouts of everything, or whether they're completely happy on getting everything off the webpage. And it seems like about 2/3 of people are perfectly happy just to deal with stuff electronically. So what we're going to do is we're going to print just a few copies of handouts, but not enough for everyone that turns up.

Okay, so I started talking a little bit about the class. I mean, in some sense this class is sort of like an AI systems class in that there's less of a pure buildup coherently theory from the ground up in a concerted way. And it's more of a class that’s built around, “Okay, there are these problems that we want to deal with in understanding natural language, and what kind of methods can we use to deal with those problems? And how can we build systems that work on it effectively.”

So there's a lot of hands-on, doing things in assignments, working out how to get things to work kind of issues, and one of the things that you’ll find doing that is often that practical issues, and working out how to define things right, and making sure that the data’s being tokenized correctly, that a lot of these things can be just as important as theoretical niceties. And I think that’s true of the field of NLP as a whole as well, that if you look at papers in the field, papers tend to emphasize their key research idea that is cool and novel, but in practice they’re the kind of systems people build if the experimental results of the papers. Yes, they contain that idea, but they also contain a lot of other hard work getting all of the details right.

And so that means that in this class we’re going to sort of assume that people have some background, and can exploit knowledge of a bit of linear algebra, a bit of probability and statistics, that they have decent programming skills. Now obviously not everyone has exactly the same background and skills. Some people know a ton about probability and machine learning, others not so much. So that we’re kind of hoping that as you go along that you can pickup things that you don’t know, are rusty on, and everyone can do well.

And I think in practice that works out fairly well because even people who’ve done a lot of machine learning often haven’t seen so much of getting things to work in practical context. And so the class tries to do a mix between teaching the theory, and actually learning techniques that can be used in robust practical systems for natural language understanding.

Okay, so where does the idea of natural language understanding come from? The idea of natural language understanding is basically is old as people thinking about computers because as soon people started thinking about computers, and thinking about robots, that they thought about wanting to communicate with them, and while the obvious way for human beings to communicate is to use language. [Video playing]

I’m sorry, Dave. I’m afraid I can’t do that.

Instructor (Christopher Manning):

Okay, so there's my little clip from Hal 2001, but I mean, actually, the idea goes much further back than that. That if you go into the earliest origins of science fiction literature that you can find in 1926, Metropolis, that you have False Maria who was a nonhuman robot, and well, not surprisingly, you could talk to False Maria. And that’s just a very natural interface modality for human beings to think about.

So in general the goal of the field of NLP is to say that computers could be a ton more useful if they could do stuff for us, and then as soon as they – you want computers to do stuff for us, well, then you notice that a lot of human communication is by means of natural language, and a lot of the information that is possessed by human beings whether it's Amazon product catalogs, or research articles that are telling you about proteins, that that’s information in natural language.

So computers could be a ton more useful if they could read our email, do our library research, chat to us, do all of these things involve dealing with natural language. But it’s precisely that point that there's this problem that, well, computers are fazed by human languages. They're pretty good at dealing with machine languages that are made for them, but human languages, not so.

Now, maybe that’s not the computer’s fault. It’s really – it’s the programmer’s fault, and partly that’s because natural languages are hard as I’ll discuss, and partly it’s because that a lot of computer scientist, and computer science think is directed in the direction that’s very different from how natural languages are. So computer scientist generally tends to want precision, or precisely specified, unambiguous, APIs and meanings for things, and natural languages aren’t like that.

So one reaction to that is to say, “Well, natural languages just aren’t the right thing for computers.” And so that’s actually been a dominant strand in the computing industry. So what people do is to find different ways of doing things whether it’s using XML, the idea of the semantic web, or designing gooies with things like menus or drop boxes, where effectively what we’re saying is, “Look. The computer just can't deal with the kind of stuff that humans produce, and how they naturally interact.”

And so instead, what we’re going to do is we're going to substitute different means of communication and interaction which are easy for computers. And well, to some extent that has been a very successful strategy because things like gooies, and XML have been very successful, but the basis of that success is essentially accepting that you can’t get computers to do clever things, but noticing that it’s very easy to get human beings to adapt and do clever things in different ways. And so we’re exploiting human cleverness rather than working out how to have computer cleverness.

And so eventually it seems like what we want to do is actually to work out some of the hard problems, and workout how we can get computers to be at the – understand and process, and produce human languages, and perhaps to learn them just as the way as two-year-old kids somehow manage to listen to this linguistic signal and learn languages. And so in this course we’ll talk about some of these open research problems in natural language.

So what kind of things do people do in natural language processing? The goals of NLP can be extremely far reaching, so there's the idea that you should just be able to pickup any piece of text and understand it. You should be able to reason about the consequences of those texts. You should be able to have real-time spoken dialogues between you and your robot.

At the other end, the goals can be very down-to-earth. So you might want to do context sensitive spelling correction. So that was picked out by Walt Mossberg in the Wall Street Journal as one of the three biggest innovations of Office 2007/2008. That’s a very simple little bit of natural language processing. You might have other low-level goals such as working out how much different products sell for at different retailers on the web.

A funny thing about the field of natural language processing is if you look how the field has moved, that in the ‘70s and the ‘80s people – there was a lot of interest in cognitive science, and people spent a lot of time working on those far reaching goals of how can we do complete text understanding? Well, if you look at natural language processing in the past decade, really, a vast amount of the work has been doing these very low-level problems like finding prices of products, and doing context sensitive spelling correction. You might think that that just means the field has been going backwards, or making no progress. I think actually it is a sign of progress because back in the ‘70s and ‘80s natural language understanding wasn’t actually practical for any real applications. Where now things are in a state where they’re actually real practical deployed applications where people can use natural language understanding, and so that’s encouraging people to work on that end of the spectrum where there are actually things that you can use.

Okay, but something I do want to emphasize about what we’re going to focus on in this course, is that we’re going focus on stuff that is in some sense going beneath the surface of the text, and involves some understanding of the text. So starting with either speech or text input. There's the most superficial level, right? So there’s an audio signal and you can record it, and you can play it back. And when you’re playing it back, you’re playing back human speech if that’s what you’ve recorded. But you’ve done no kind of processing or understanding of any of that.

Similarly, for text format input, you can do this very superficial level of processing where you cut your words and white space, and then you maybe index them in the information retrieval system like the very kind of base level of web search, and then you just kind of match words, and you give stuff back. And again, it seems like you’re not really doing any understanding of natural language. And while that’s in some sense the very lowest level of natural language technologies, I’m really going to be interested in this talk, and how can we start to do things which don’t fully understand the text, but somehow go beneath the surface and start to recognize certain parts of the structure and meaning of human language text, and can do cleverer things because of that. And you start to see that happening even these days in even sort of effectively base the whole technologies like information retrieval, and web search, that once upon a time you just cut in the key words, and you index those key word. Where now all the search engines are increasingly starting to do things like, how can we do things like having different forms of the word count as equivalent? How can we do query expansion to find or turn [inaudible] related terms, and things of that sort.

Okay, well, what is it that’s holding back the field of natural language understanding? Here’s a quote that I’ve always rather liked. It’s from Bill Gates who’s actually always been an extremely big supporter of natural language understanding as part of his general interest in using natural language interface technologies. So Bill Gates says, “Applications always become more demanding. Until the computer can speak to you in perfect English, and understand everything you say to it, and learn in the same way that an assistant would learn, until it has the power to do that, we need all the cycles. We need to be optimized to do the best we can. Right now, linguistics are right on the edge of what the process it can do. As we get another factor of 2, then speech will start to be on the edge of what it can do.”

Well, that sounds a nice story, and the story that’s being told as well, just give us a few more CPU cycles, and well, everything will work fine. But the problem is if you then look up at the top of the slide, that these are actually remarks that Bill Gates made in 1997. And if you can remember back to 1997, I think that was maybe when what people had was original Pentiums that ran at 166 megahertz, or maybe Pentium Pros at 200 megahertz had come out. I mean, either way you look at it, since that time computers have gotten an order of magnitude faster in terms of their CPUs, and we now have our dual core, and quad core TIFFs, and all of these things. So it's sort of fairly easy to be getting two orders of magnitude of performance compared to what we had then. But somehow, that hasn’t solved all of the problems of natural language processing. It seems like – well, oftentimes it still seems like we’re just on the edge of what computer technology can do. That natural language processing, and speech recognitions sort of work, but still don’t work quite well enough to satisfy what people need out of them. And so I think one of the important things to realize from both this lecture and the class at a whole, is just to really actually get some sense of why this problem is hard. I mean, some things we can do really well with computers these days. Right, the visual effects that you see in your Hollywood videos are completely amazing, and amazingly realistic, and lifelike. So why isn’t our ability to do human language understanding equally as good? And I think one way to think about that is to think about the history of NLP, what kind of computers people had and where things headed from that.

So if you look at the early history of NLP, NLP essentially started in the 1950s. It started just after World War II in the beginning of the Cold War. And what NLP started off as is the field of machine translation, of can you use computers to translate automatically from one language to another language? Something that’s been noticed about the field of computing actually is that you can tell the really old parts of computing because the old parts of computer science are the ones that have machine in the name.

So that the association for computing machinery is a really old organization and you have finite state machines. They were things that were invented a long time ago, and machine translation was something that was studied a long time ago. Whereas something like computer vision, or graphics, there's no machine in the name because really those were fields that didn’t even begin until decades later.

So the beginning of NLP was essentially in this Cold War context where the Soviet Union and the United States were both convinced that the others had scientific advances that would allow them to dominate the world. And the problem was they wrote about them all in their own respective languages, and so both the Soviet Union and the United

States started pouring huge amounts of money into this goal of machine translation to be able to translate from one language to the other language.

And at the beginning, there was a lot of excitement, and I’ll show you that in just a moment. But that excitement soon turned to disappointment. Maybe I should show you the excitement first. [Video playing]

So this is a video from the 1950s talking about early work on machine translations.

Instructor (Christopher Manning):

So I’ll stop there, and I’ll talk about why it runs into trouble in a moment in the context of natural language understanding. But I mean, in retrospect it's not very surprising that this early work in the 1950s didn’t go a long way. When you think of the kind of computers were available in those days, that they had far less computing power than what you now have in your cell phone, or even your average pocket calculator. And so one part of it was that there was no – there was really just about no computing power. But the other part of it is that there was almost no understanding about human – how human languages worked. It was really only in the late 1950s and early 1960s that work started to be done on understanding how the structure of human languages in terms of how words put together in sentences really worked.

So if people in computer science know the name Noam Chomsky, they generally know it for two reasons. One reason is because of his political take on the world which I’m not going to discuss here, and the other reason is because of the Chomsky Hierarchy. And so the Chomsky Hierarchy gets taught as an understanding of levels of complexity of formal languages, and that’s related to things like building transducers, and compilers, and things like that.

But Chomsky didn’t actually develop the Chomsky Hierarchy for the purpose of making your life grim, and formal languages, and altimeter classes. The reasons that Chomsky Hierarchy was developed was that so that he could try and get a handle on, and present arguments about what the complexity of the structure of human languages was. That he was wanting to argue that the structure of human languages was clearly beyond that of context free languages, whereas most of the models of language that were actually being built in the ‘30s, ‘40s, and early ‘50s were actually modeling language as finite state, and was there for [inaudible] or completely inadequate the complexity of human languages.

And so it was really just – there was neither the science nor the technology for this work to be able to go far. So if you look at what was actually produced, what there was was essentially word lookup and substitution programs. So you’d take your sentence in Russian, you’d look up the words in the dictionary, and replace them with the first word that you saw in the dictionary as a possible translation equivalent. And as we’ll see when we start talking about – more about machine translation later today, that that doest work very well. There’s a lot more going on in languages that make them difficult.

And so after awhile people noticed that the problem seemed intractable, and so there was this famous report that was issued by the U.S. government in the 1960s, the ALPEC report which essentially cancelled all work on machine translation research in the United States. I think rightly arguing that given the technology and science that existed at the time, that this was just a hopeless endeavor and essentially recommending that what instead should be being done is the promotion of basic science so that people actually had a much better understanding of how human languages worked.

Okay, so let me go on for a little into then talking about this question of why natural language understanding is difficult, and essentially the point that I want to bring home is that natural language understanding is in the space of what sometimes people call inverse problems. And the idea of inverse problems is that there's stuff that you see on the surface, and then you have to construct stuff that underlies it, and doing that is always the hard problem.

So the analogy is to go back to my example with movies so that there are two twin problems. There's computer graphics where you want to generate good graphics of a scene, and there's computer vision where you actually want to be able to look at a scene in a photo, and understand what’s in it as a three dimensional model in your head.

So computer graphics is the direct problem. That you have a model of what you want on the screen, and then you realize that in graphics. And direct problems are the easy problems. Computer vision, that’s the inverse problem where you’re just seeing the surface picture and you want to reconstruct what objects there are in that scene. And that’s still a really difficult problem. And in just the same way, natural language understanding is an inverse problem. That you're going from a speech signal, or sequence of words, and you’re trying to reconstruct the structure of the sentence and the meaning of the sentence that underlies it. And the reason that that’s difficult is that natural languages are extremely, extremely ambiguous. So here’s a fairly straightforward sentence. My example here is a few years old, but ever year the Fed is either raising or lowering rates. So we have this sentence, “Fed raises interest rates half a percent in effort to control inflation.”

Well, you know, that’s just a sort of an obvious sentence. That’s the one time, if you look at the financial section of your newspaper that’s kind of one you read every year. It's not hard to understand. For a human being it doesn’t seem like there's any problem there. That it doesn’t seem like there's any ambiguity at all if you just read it casually. But if you actually start looking at the sentence more carefully, there are tons and tons of ambiguities. So there's the word “rates.” That can be either a noun or a verb, so it can be, “She rates highly,” or “Our water rates are high.” Noun or verb. The word “interest,” that can be a noun or a verb. You can say, “Japanese movies interest me,” verb. Or you can say, “The interest rate is 8 percent,” noun. “Raises,” that can be a noun or a verb as well. So it can be “Fed raises,” is a verb, or “The raises we received was small.” That can be a noun. So you notice this fundamental fact that in English a ton of words can be nouns or verbs. I mean, and very generally you can take nouns and turn them into verbs. So you have a noun like butter and you can turn that into butter your bread. You have a noun that’s a protocol like SSH, and then you can immediately turn that into a verb, and say, “I SSH’d the file to you.”

So that kind of already introduces a ton of ambiguity. And so what does that mean? Well, what that means is that there are a ton of different syntactic structures that we can build because of that noun/verb ambiguity. So if you combine the two facts that raises interest and rates can be nouns and verbs, and the fact that you can make noun compounds in English by just sticking nouns together, you get a lot of ambiguity. So noun compounding is things like computer network where you just stick two nouns next to each other to build a bigger noun. Very productive in the computer industry. That’s why when you read the manuals and you have this sort of personal computer, power supply, enclosure, that’s just all of the compound noun where you’re sticking nouns together.

So here’s the structure that we meant to be getting where we make interest rates the compound noun in Fed raises interest rates, that combining those two facts together you could instead make Fed raises a noun phrase as a compound noun, and then interest rates where interest is the verb. Or you could have Fed raises interest as the noun phrase, and rates as the verb. And not only are these sentences syntactically wellformed, you can tell a sort of semantic story for them as well. It’s not that you can’t come up with a semantic translation for these.

So this is the normal one where the Fed is raising interest rates. For this one, I could tell a story about that there was an article in yesterday’s paper that the Federal Reserve Board had decided to award everyone on the Board, i.e. themselves, a large pay raise and then you could have a headline that says, “Fed Raises Interest Rates,” meaning that the market rates were going down, or up because people though policy on the Fed was lacks, or the Fed was corrupt, or something like that. And then once you have that idea, you could go to this one, and now we’ve got something like the Nelson Ratings of television, and one of the stations was showing a news item about these Fed raises. And so Fed raises interest is then the level of interest in rate – in raises at the Fed, and that rate’s only half a percent viewer ship, almost no one watched it.

So it’s a bit of a stretch, but it’s not that you can’t tell a semantic story about those – how those fit together as well. So those are all the kind of ambiguities that our computer systems will find. Somehow as human beings, that even as we just read or hear the words we’ve got all of this kind of highly contextually based plausibility deciding going on in our heads at the same time. So that most of the most we just never notice those ambiguities, we just kind of go straight to the most plausible interpretation and stick with that.

So that’s one kind of ambiguity. There are lots of other kinds of ambiguity, so just to mention a few more. At the word level it gets still, as well as having these syntax ambiguities of noun or verb, you also get semantic ambiguities, so called word sense ambiguities. So the word Fed here refers to the Federal Reserve Board, but the word Fed is also used for an FBI agent, so “A Fed trailed me on my way home.” It can also of course be the verb that’s the past tense of the verb to feed. Interest, that’s a word with tons of meaning, so it can mean something that you like. “My main interest is collecting mushrooms.” It can be that you have a stake in a company, “He has a 5 percent interest in Vulcan holdings.” It can be a rate that you're paid, the 8 percent interest rate, lots and lots of meanings for the word interest.

But then there are also ambiguities above the word level. So towards the end of the sentence you have these phrases “in effort,” and “control inflation.” So “in effort” is a prepositional phrase, then you have this infinitival phrase “to control inflation.” Whenever you have these kinds of prepositional phrases, or infinitival phrases, you have ambiguities as to what they modify. So is “in effort” going to modify this noun phrase of half a percent? Or is it going to be modifying the verb “raises” and we’ll see some more ambiguities of that kind later.

But to give us slightly more of a sense of what kind of ambiguities occurs, let me show the funny examples. These are all examples of real newspaper headlines which have ambiguities. So Ban on Nude Dancing on Governor’s Desk. What’s this one? Well, this one is precisely one of these prepositional phrase attachment ambiguities. So we have the prepositional phrase, “On Governor’s Desk,” and there are two ways that you can interpret that. You can take “On Governor’s Desk” as modifying “Nude Dancing.” So then you have, “Nude Dancing on Governor’s Desk” as a bigger noun phrase, and there’s a ban on that. So that’s the funny reading.

Then the reading that you're meant to get is that there's a ban on nude dancing, and that ban is on the governor’s desk. So that what you're effectively then have is this prepositional phrase “On Governor’s Desk” is actually modifying the word “Ban.” And so that’s refereed to in a lot of the literatures where does the prepositional phrase attach? Meaning, what does it modify?

This one’s dated a few years, and so I guess this one comes from prior to 1991. Iraqi Head Seeks Arms. Like, so this one is then playing on words sense ambiguity. So “Arms” can be real arms of animals, but here it’s talking about weapons. And “Head” can be a real head, but here it’s then referring to a head of state. In both those usages are affectively illustrating the way in which a lot of new words senses come into being. That word senses often come from what are referred to as sense extensions which occur fairly productively where more or less metaphorically people will be extending the meaning of one word into further domain. So that you have the idea of a head on a person that’s controls the person, and so you then extend that idea to a state or a country, and then you have a head of the country who is the person who controls that. And that keeps on happening all the time. So you notice all through recent developments in the computer industry. So that when we have the web, and then you spider the web, and all those things, it’s all metaphorical extensions that are going on.

Juvenile Court to Try Shooting Defender. This is then again, an ambiguity of syntactic structure in terms of what’s modifying what. So one possibility is that you have shooting as a modifier of defendant, so you have a shooting defendant, and then you put together the rest of the sentence so that “to try” is then the main verb, and you’re trying the shooting defendant. But the alternative is that you have shooting as a verb, and defendant as the object of that verb. So you're shooting the defendant, and that’s something the court’s going to try.

And then we can continue on from there, and there are a bunch of other examples. Teachers Strikes Idle Kids; Stolen Painting Found by Tree. This is more sort of semantic interpretation of “by tree” as the to whether that’s being regarded as the agent of the finding, or the location of the finding. Local High School Dropouts Cut in Half; Red Tape Holds Up New Bridges; Clinton Wins on Budget, But More Lies Ahead; Hospitals are Sued by Seven Foot Doctors; Kids Make Nutritious Snacks; Minister Accused of Having Eight Wives in Jail.

So these are the sort of funny examples where there are real ambiguities that human beings notice, and in a sense I went looking for these in a place where you can find them because there's a fact about the way that headlines are constructed that makes these kind of funny ambiguities more likely to arise, and that in headlines you often leave out a lot of the function words, and so that tends to increase the amount ambiguity that’s going on. So in some sense this is atypical, but I think the example that you more want to still keep in your head is that example of Fed Raises Interest Rates Half a Percent in Effort to Control Inflation. I mean, that example’s not a funny example. When human beings read it they don’t see any ambiguities at all. And yet still in that example there are tons of ambiguities present. And I’ll show just one more example that then relates too where there are semantic ambiguities. So in the old days in the GRE, and still in the LSAT they have these logic puzzle questions. And I think probably often the kind of people have become computer science majors that probably some time in high school or something you’ve seen these kind of logic puzzles.

So the nature of the questions is that there's a scenario and then you’re meant to answer a question. So six sculptures are to be exhibited in three rooms, and then there are some conditions. Sculpture C and E may not be exhibited in the same room. Sculptures D and G must be exhibited in the same room. If sculptures E and F are exhibited in the same room, no other sculpture may be exhibited in that room. And then there's a question that you're meant to ask.

Now because these are [inaudible] problems that determine whether people get into graduate school, that the people who build these at ETS they’re whole desire is to make these questions completely unambiguous, so that people don’t sue the ETS saying that they couldn’t get into grad school successfully. But it turns out that the human languages; you just can’t make them unambiguous because just by nature natural languages are ambiguous. So you take a sentence like this last one, at least one sculpture must be exhibited in each room, and no more than three sculptures may be exhibited in any room. So these phrases like at least one sculpture, in each room, no more than three, any room, those are what are referred to as quantifiers. And it's just a fact about English that whenever you have multiple quantifiers you have scope ambiguities. So there are different ways to interpret things.

So if we take this last bit, no more than three sculptures may be exhibited in any room, the way you're meant to interpret that is for any room it has at most three sculptures in that. But it’s also possible to interpret things differently with a opposite order of quantifiers. So you could have the interpretation that there's a special set of sculptures which numbers no more than three, maybe they’re ones that the museum owns or something, or aren’t precious, and these ones are capable of being exhibited in any room in the museum. Whereas all the other sculptures somehow there are restrictions in which rooms you're allowed to show them in. But that’s again, a perfectly good semantic translation according to how natural languages work.

Okay, I have one more example of that, but maybe I should start to get ahead and so I get to say a bit of stuff about machine translation before we run out of time. But the lesson to takeaway is that natural languages are highly ambiguous at all levels. That the reason that they work for human beings is because natural language understanding essentially depends on making complex and subtle use of context, both the context of the rest of the sentence, and context of prior discourse and thing around you in the world. So for humans it’s obvious what people are saying.

But that means that it’s hard for computers to do things, and so somehow there's a lot of different factors that influence how things are interpreted. And so that means that – that suggests this approach where we want to somehow make use of probabilistic factors and reasoning to try and do a good job of simulating how human beings can understand natural language.

So [inaudible] doing natural language understanding is extremely, extremely difficult. It’s sometimes referred to as one of the AI complete tasks. But it’s also important to counter that with the other extreme, which is it turns out that there’s some other things that you can do with NLP which turn out to be surprisingly easy and to work rather well. And so a lot of the time we have a vast amount to text available to us, and we can do a surprisingly good job for many tasks by just looking through a large amount of text and counting things, and then predicting basic on counting. So that’s the kind of thing that’s done with context sensitive spelling correction. You just count up how often different spelling occur in different context, and you say that’s the way it should be spelled when a gap appears in similar context. And ideas like that just work incredibly successfully, and are incredibly useful to people. So somehow we want to be in this middle ground where we’re doing things that are sometimes not even that hard, but give useful results.

Okay, and so that’s led into this direction of how can we do a fairly robust and reliable way of combining at least an approximation of knowledge about language in terms of how language is used in other context. What kind of context we’re in at the moment, and put that information together to come up with the right interpretation of sentences in a straightforward way. And the answer that’s been getting a lot of traction in the last decade and a half or so in natural language processing is to be using probabilistic models. It’s essentially the whole of probability theory, the idea of it is, well, we have uncertain knowledge about the world, but we want to combine all the evidence that we have which we may in turn the uncertain [inaudible] and solve it, and combine it together to give the best prediction or understanding we can of what is intended, and what was said, et cetera. That’s exactly what probability theory is for, and it works rather nicely for a lot of natural language problems. So that the general idea is that anything – it doesn’t immediately look like natural languages look like things you normally see in probability theory because when you see probability theory you normally see numbers, and normal distributions, and things like that. But we’re going to imply just exactly the same ideas to language stuff so that we have the French word [inaudible] which might be translated as house, and we to just learn the fact that that’s a high probability translation. Whereas we have [inaudible], general, we don’t want to translate that as the general avocado, but we want to say that we’ve got enough context around the word avacard that we can see that translating as avocado would be a very bad idea.

Lately using probabilistic models has become quite trendy and widely used in a lot of areas of artificial intelligence. You see it in vision, you see it in machine learning, you see it in various forms of uncertain knowledge representation. That’s it’s sort of just slightly intrinsic historically that the use of probabilities in natural language understanding didn’t come from those recent developments in artificial intelligence. Where this strand of work came from is actually from electrical engineering. So in the greater field of natural language there's text or natural language understanding, and there’s speech recognition. And those two sub areas haven’t always had as much communication between each other as they should have had. And departmentally speech has always been mainly based in electrical engineering departments.

But what happened was that at the same time that all the people in computer science and AI in particular were convinced in the use of symbolic methods, and doing logical approaches to artificial intelligence, and things like that, the speech people, often electrical engineering departments, were separated enough off from the kind of things that they were studying was that they were still studying how to use probabilistic methods. How to use real numbers and integrals for doing signal processing, and all of those kind of things. So that the natural way for electrical engineer and speech people, the process speech recognition was thinking about in terms of probabilistic models of reconstructing words from an uncertain audio signal. And some of those people then started thinking about how could we then start applying some of those ideas further up the feeding chain for doing other kinds of natural language processing. So maybe we could use those same probabilistic ideas to work out the part of speech of words. And essentially this work really came out of two places. It came out of AT&T labs, Bell labs, in those days, and IBM research on the East Coast. That there were groups there that started spreading from speech into using probabilistic techniques for natural language understanding, and essentially that lead to natural language processing being revolutionized by probabilistic methods earlier than, and quite separately from any of the work that happened in using probabilistic methods in other parts of AI.

Okay, so in the rest of this course what we’re going to do is look at a bunch of different problems and natural language processing, and how they can be approached.

So the very first problem we're going to start off with is that first problem of NLP machine translation, and looking at how statistical machine translation systems are built. And I'm going to use that as a backdrop for introducing some of the other key technologies we need for using probabilistic models in NLP such as building language models, and algorithms for estimation, and smoothing, and things like that.

I'll then go on and start talking about sequence models, the task like information extraction, and part of speech tagging, and then move on to doing natural language parsing, looking context free grammas and probabilistic equivalence of those. And then after that we'll then go further into doing semantic interpretations, the text, and particular applications.

There's only time in the quarter to look at a few applications, so there are lots of other things that I'm just going to have to leave out, but try and do a few things in enough detail that you have a kind of a good understanding of the main tools that people are using these days for natural language processing.

Okay, so the first problem that we're going to be starting with is this problem of machine translation. So the idea is we're going to start with this piece of text here, which most of us aren't very good at reading [inaudible] there were these characters. [Inaudible] at the beginning, but my Chinese isn't very good, and we'd like to turn it into this text where we can just understand and say, "Here's the U.S. island of Guam is maintaining," but many people have thought of this as kind of the classic acid test for natural language processing. But if we could have a system that could fluently translate text that means that we have completely solved the problem natural language understanding.

I think in modern times people are actually a little bit less convinced of that because although one approach to doing natural machine translation is to do full text understanding and presumably, that's essentially when a human translator does most of the time. Really if you look at just the inputs and outputs of machine translation, well, both the input and the output are streams. So really, machine translation is just a string transduction problem, and it's quite possible that you could do that quite well without really understanding all of the text. And indeed quite a lot of the work on statistical machine translation has taken exactly that approach.

But nevertheless, regardless of whether you see it as the fundamental test of natural language processing, it's clearly an immensely important practical problem. You see it a little bit less in the United States than in some other places since English is so dominant in much of the United States, but if you look at other places like the European Union, so the European Union spends billions of dollars each year translating stuff. Even in the United States if you're a company that does business internationally, you have to pay to translate stuff. The U.N. spends an huge amount of money translating stuff. And so here's just a cute little example that I got the other week from Scott Klemmer which shows that how even kind of modern, new age web companies can't escape from the cost of translating stuff. So Scott was talking to some of the people at Google about Google SketchUp, and they pointed out this fact that while they only release a new version

Google SketchUp every 18 months. Why do they do that? It’s not because they’re not working on the code, and couldn’t regularly release new versions of the code. The problem is that they can’t get the translations of the user manual done more regularly than that.

So it's really doing machine translation – doing translation that is really killing a lot of the sort of productivity for being able to be nimble.

Okay, here’s a few examples just to give you a sense of the kind of stuff that happens in translation. So there's another piece of Chinese text and the reference translation is, according to the data provided today by the Ministry of Foreign Trade and Economic cooperation, as of November this year, China has actually utilitized $46.959 billion U.S. dollars of foreign capital, et cetera.

The main thing that we’re going to talk about, and that you guys are going implement some of for the second assignment, is these models – the canonical early models of machine translation by statistical means were proposed by IBM, and they built five models of increasing complexity called IBM Models 1, 2, 3, 4, and 5, not very original names. And for the assignment we’re going to get you guys to build Models 1 and 2 because you can implement Models 1 and 2 fairly straightforwardly where things get much more complex if you go up to Model 3. But here’s some output from IBM Model 4. The Ministry of Foreign Trade and Economic Cooperation including foreign direct investment $40.007 billion U.S. dollars today provide data include that year to November, China actually using.

Well, there are some snatches of that in particular some noun phrases that they system just did fine, right. The Ministry of Foreign Trade and Economic Cooperation, that’s fine. Then the $40.007 billion U.S. dollars, somehow the number came out slightly different, but maybe that’s exchange rate. But that’s kind of an okay bit, but clearly if you then look at other parts of the syntactic structure it’s just completely garbled because the syntactic structure of Chinese is rather different to English.

So when you have these bits like, U.S. dollars today provide data include that year to November, China actually using, right, the [inaudible] it seems like most of the words are there, that you’re kind of looking up words and sticking the translation in just like people were doing in the 1950s. Here it’s being done with somewhat better use of context, so you're choosing the right words out of ambiguous translations. But clearly the syntax is just kind of completely garbled, and it's not making any sense.

So when you do the assignment don’t expect your translations to be perfect because really they’ll be worse than that one probably. But then in the bottom – towards to the end about some of the more recent work of trying to syntax based machine translation models which then do a better job at maintaining the struc – while recognizing therefore correctly translating the structure of sentences. And so this is work from [inaudible] and their translation is, that today’s available data of the Ministry of Foreign Trade and

Economic Cooperation shows that China’s actual utilization of November of this year will include $40.07 billion U.S. dollars for the foreign directive investment.

I mean, it’s not quite perfect. It's a little bit awkward in a couple of places, but you can see how the syntactic structure of the translation is just a ton better because it’s been correctly recognized in the source. And it's actually starting to be quite a passable translation.

Okay, so this is a very quick history of machine translation research. I’ve already mentioned the work in the 1950s and ‘60s, essentially just direct word for word replacement. That was then the kind of shutdown of empty funding and research in the U.S. following the ALPEC report in 1966. There then started to be a resurgence in other parts of the world, maybe parts of the world that needed translation more. So in Europe and Japan these were the days of the Japanese Generation Project when they were going to take over the world with UAI. But that’s sort of started to slowly lead in the late ‘80s and the early ‘90s to a gradual resurgence of machine translation work in the U.S. But what really turned the world around was when these IBM machine translation models were produced, and the approach of statistical machine translation started to be hotly pursued.

So over the time that I’ve been doing this job, things have just been completely turned around where from in the late 1990s almost no one worked in machine translation, and most of the interesting citing research work was being done in the other areas of natural language processing like parsing information extraction, and various other topics. Whereas really in the last five years machine translation and a particular statistical machine translation, has just become the hot area of machine translation. So these days all of the young graduate students, or assistant professors when they get jobs, it seems like fully half of them what they want to be doing is doing statistical machine translation. And so there's a ton of new and interesting work leading to much better statistical machine translation systems.

Okay, so what happened between ALPEC when it was deemed impossible and now? Well, a lot of things have happened. The need for machine translation certainly didn’t go away. In fact, the need for machine translation is just increased and increased for various kind of economic and political reasons. So the greater internationalization of trade, and multinational companies, the greater unification that occurs through things through the European Union, or Hong Kong becoming part of China. That everywhere that there's sort of more and more need for translation.

The technology has gotten a ton better, and that’s made things a lot more possible. The other big thing that’s made things a lot more possible is not just the computers, but actually having the data. In the early days that there just wasn’t online digital data in large amounts to do thing with, whereas now we have a ton of such data the most obvious part of which is looking at the World Wide Web, but just more generally all of the data that’s available in digital form. That there are now just literally billions of words of data in major languages that anyone can get their hands on. Of course now we also know more about linguistic structure and that helps.

The final thing that has helped progress is this item that I've got down there as change in expectations. That the change of expectations part is saying, there's something interesting that's happened is that there's – become new opportunities for kind of so-so translation. That in the early days essentially the only use for translation was for producers of documents. So people had their user manual and they wanted to translate it into Spanish, German, French, Italian, et cetera. And by and large those people had to pay human beings because machine translation was too bad for them to be able to release on the market that their product because frankly it would look bad.

Notwithstanding, those consumer electronic products where even when the human beings translate them, that commonly the translation is still really bad, and makes the product look bad. But the thing that's changed now is that we now have much more of a system through things like the World Wide Web where there are users that can originate their own translation. So that I might want to do something like go and look at the Al Jazeera webpage and wonder what it's saying, and my Arabic isn't very good, so I want it translated for me. And I'll be happy providing can I understand the gist of what the page says. I don't really mind if the translation isn't perfect. It'll be a lot better than having nothing. And that's precisely why there's now this opportunity that you find on all the major search engines like Google and Yahoo, and Microsoft Live Search web, they can do these low quality translations for you. And that same facility is being used by users in lots of other places as well. So everything [inaudible] doing online chat, and playing games, and things like that, that you can help out billions of people who's primarily language isn't English especially by providing these kind of facilities.

Okay, so overall there are different approaches to machine translation based on how deep you are going in attempting to translate. And this was presented a very, very long time ago by someone Voqua, which is then called the Voqua Triangle. And the idea here is we've got the source text, and we've got the target text. And while we're able to do some amount of processing, and then at some point we're going to try and convert across to the other language.

So one possibility is that we do extremely minimal processing, we presumably want to cut the text into words. We might want to normalize the words a little, and then go straight at the word level across the target language. And so that's then called direct translation.

That was what was done in the 1950s when statistical machine translation started in the 1990s. Actually, that was again what people did. It's just that they were doing it with statistics to try and do it better. But most people have believed that ultimately to get better translation you're actually going to want to understand more of the source text to use that information to make a better translation.

So that you might want to do syntactic analysis produces syntactic structure for the sentence. And then you could do syntactic transfer to a syntactic tree in the other language, and then do generation from there to the target text. And that’s the kind of area that’s sort of really the research area these days, at how to do syntax based statistical machine translation.

But you could want to, and there was work in the 1980s especially that did go even deeper than that. You might want to go up to the level of having semantic analysis of a particular language, and then do semantic transfer. And then people have had the idea, and again, this was sort of worked on actively in the 1980s, that maybe we could go even deeper than that. We could kind of go beyond source language, whatever it is the Chinese semantic structure. Why can’t we define a universal semantic structure, which could be used for all languages? Which is then referred to as the interlingua. And so it’s that interlingua-based machine translation where you translate from one language into lingua, and then from the interlingua back to the other language.

There’s an enormous appeal of that approach because if you could actually do it then you can handle all of the languages of the world without very much work. Because if you were down here, or down here, and you want to translate it between a lot of languages, the problem is if there are end languages that you wanted to cover you have to build end squared systems. And so that’s the kind of problem that Google still has today, right? That you can pull down the menu of what languages you can do translations from and they’re building systems for individual language pairs. So you can translate from Chinese to English, or English to Chinese, and French to English, and English to French. But if you want to translate from Chinese to French, it doesn’t provide that. So you're in this sort of bad end squared space.

Whereas if you could actually do this generation up to an interlingua then you’d only have to build order end translation systems, and you could translate between every language pair. So that’s seems very appealing, but on the other hands it’s been a very hard idea to workout. And this is perhaps a false idea because the problem is individual different languages have all kinds of distinctions, and ways of looking at the world that are of their own and aren’t reproduced in other languages. So you know, there are particular things that we make distinctions about in English. We distinguish peas and beans as separate words. Whereas in most language don’t actually distinguish peas and beans, right? They could have words for different kinds of beans, same as we have words for farmer beans, and lima beans, but that they’re different kinds of beans. And every language does that, right? This is kind of like the Eskimos having a lot of words for snow. And the problem is if you're actually trying to design an interlingua that works for every language, that means you have to put into the interlingua the complexities of every individual language, and that makes it a very hard goal to get to.

Okay, so we’re not going to be heading in that direction. But we’re going to be looking at how statistical machine translation can be done at essentially the word direct translation level, and then talking a little bit about how to do it at the syntax level.

And so I'll have you do machine translation. The essential answer to how to do machine translation by statistical methods is this picture. So this is a picture of the Rosetta Stone which is the famous basis for how the decoding of hieroglyphics happened. And well, how did that happen? How that happened was because someone had a hunch that this stone contained the same text in three languages, in Greek, in Demotic Coptic, and Hieroglyphs, and this assumption that this text, this correct assumption, that this stone had the same text in three languages gave enough information to start to then workout how you could decode hieroglyphs.

And so this is exactly what we're going to use. We're not exactly going to be decoding in exactly the same sense, but we're going to use the same source of information. We're going to take text that we know is parallel. There's lot of sources of parallel data these days in the world because people like the European Union translates stuff, or the Hong Kong Parliament produces proceedings in Chinese and English. And we're going to exploit this parallel text to induce statistical models of machine translation. So here's a list of just some of the places where you can get parallel text. There are lots of others as well, so lots of magazines appear in multiple languages.

And so what we're going to do is have our computer look at lots of this parallel text, and say, "Hmm, every time I see a particular word in the Spanish text, the translation of this sentence has particular words in the English text, and so that will be my translation."

This is actually in a sense also old idea because I mentioned already that in the early years after World War II, a ton of money was thrown into machine translation research. The person who actually initiated that work was Warren Weaver who was a prominent person in the U.S. government in the years after World War II.

So Warren Weaver essentially was the person that initiated this research in machine translation, and precisely where his thinking came from was coming from the direction of knowing about code breaking. And so Weaver got interested in where that idea could be exploited because he had noted the fact that during World War II that what were computers used for in World War II? Essentially, they were used for working out the flight path of bombs, and for code breaking.

And so thinking about machine translation he got interested in can computers do translation as code breaking? And he wrote to a professor that he knew who was a foreign languages professor about this idea. And he wrote, "Also knowing nothing official about, but having guessed and inferred considerable about the powerful new mechanized methods, and cryptography. Methods which I believe succeed even when one does not know what language is being coded. One naturally wonders if the problem of translation could conceivably be treated as a problem in cryptography.

When I looked at the article in Russian, I say, "This is really written in English but it has been coded in some strange symbols. I will now proceed to decode." So that was – seemed to be the idea in Warren Weaver's head.

Now in practice in terms of what was actually built in the 1950s and ‘60s, really no one picked up on this idea at all. That people started hand building translation dictionaries, and hand building analysis rules in grammas, and hand building a big system, and it wasn’t approached in this manner at all. So strictly it was then in the ‘90s that – well, late ‘80s, ‘90s that statistical work picked up this idea again, say, “Well, let’s start with this parallel text and view language translation as a decoding process. And let’s see if we can just automatically get our computers to build models of the translation process.”

And so in a couple of lectures, I’m going to go through the details of how the IMB models started to do that. In the very next lecture, I’ll first of all talk about language models which is a key ingredient we’ll need along the way. But for the last five minutes of this lecture, what I’d like to do is sort of illustrate this process by just showing how it works at a human level working with a small simulated parallel text.

And so this is following a kind of a nice example that was done by Kevin Knight who said, “ISI at the University of Southern California, and ISI has been one of the most prominent centers where people have worked on statistical machine translation in the last decade.”

So the Kevin Knight scenario is what we want to do is translate between Sentory or from Sentory into Aucturan. So that we start off with this sentence in Sentory [foreign language] and somehow that we want to learn how to translate this into Aucturan.

Well, fortunately for us we’ve actually also managed to find a little bit of parallel text. Some example of sentences in Sentory for which we know their translation in Aucturan. So here’s our parallel corpus. We’ve got these 12 pairs of sentences in the two languages.

Now obviously when we ask our computers to do this task typically – well, if we’re building large systems commonly we’re actually giving them millions of words of parallel text, and even [inaudible] system we might begin at tens of hundreds of thousands of words of parallel text. But here I’ll just illustrate it with this teeny, teeny text. But nevertheless, the – although we’ll do it here as human reasoning, what we’re doing is essentially exactly what we’ll want to build an algorithm to simulate.

So what we’re going to build for algorithms is what are called EM based alignment algorithms. So I’ll talk later about the EM algorithm. It's a general probabilistic estimation algorithm which tries to learn how to do translations. It learns alignments which are parings between translated words.

So how might we do it in this context? Well, an obvious thing to do is to start with the first word in the Sentory sentence. So suppose we look at that word furock. Well, what would furock mean? Well, essentially our source of evidence is to say, “Well, can we find furock in our parallel corpus? Oh yes, furock appears twice in our parallel corpus.”

Well, if we don’t know anything else a reasonable thing to assume is maybe that one of these words translates the word furock. And presumably that’s true here. So a good heuristic would be to look for a word that appears in both of the translations and while there's precisely one of those words, this word ja.

Another heuristic that we might have used is is we might have guessed that word order between the two languages has some similarities. So the fact that furock is second in both of these sentences and ja is second in these sentences too, that’s at least weak confirmatory evidence. And so tentatively we could say, “Well, we can translate furock as ja.”

Okay, so then we can go on to the second word, crok. That’s a kind of a tricky one because crok only appears once in our parallel corpus, and well, I mean, if we really believe word order was always the same we could count, but that seems kind of weak evidence since we know word order varies a lot between languages. So that one seems kind of too hard. So maybe we should just leave it for a moment and say it could be any of those words in the translation, and maybe we could go on and look at other word.

Okay, so hehawk looks rather more hopeful because we can find hehawk in three sentences. And so again, we could do this process of saying, “Okay, which word – well, if we look at these three sentences it seems like arrat appears in all of them.”

Now you know, on practice a lot of time we won’t be this lucky because a word in Sentorian will be translated sometime by one word in Aucturan, and sometimes by a different word. And so then it’s a bit harder, but we’ll at least look for words that frequently occur in the translation of the sentence.

But here we seem to be in good business, and we can go with that alignment. And so we can move onto yurock. So we can do the same kind of reasoning and say, “Well for yurock that seems quite plausible that we can line that with maut. And so that then gets us onto this word of clock. So clock is again the tricky one because clock only appears once in the corpus. So we’re still in this – we might think that we’re in this situation, “Well, it only appears once. It could be any of those words in the other sentence. We can’t make any progress.”

But actually that’s no longer true because since we’ve been kind of working hard on this problem, we can start to make some progress now because although if you just look at this you could think it could be anything. Well, we’ve been looking at various words, and so we can start to work out by some kind of process of elimination what might be going on.

So that if we realize that – well, actually we’ve worked out good alignments for other things, that these are good alignments. Well, now it looks like, well, clock is the only word left over and ba hasn’t been aligned to anything, so maybe what we should do is translate clock as ba. And so that’s this – been giving us a process of elimination where exploiting of some words we know we can guess that the remaining words go to untranslated words. And the EM algorithm, one of the features of it is it provides an implementation of this sort of process of elimination idea.

There are other sources of information we can also use for helping us along in some cases. So if we look at this next sentence down here, we've got two words that we haven't aligned here. One was the crok that we wanted to work out at the beginning, and then we have zanszenot. So even using the process of elimination idea, there are still two words here and one word there, and it might not seem quite obvious what to do. But a possible source of information is to see that zanszenot and zanszena. Well, they look very similar to each other, and so often words that look very similar in different languages actually are related to each other. Those words are referred to as cognates.

And now for some languages which are related historically, or had a lot to do with each other for various kind of historical reasons, there are tons of cognates. So French and English have tons of cognates, and then as related in the European languages, and even for languages which aren't actually related closely in the language families for a kind of historical and cultural reasons there maybe a lot of borrowed words that give cognate.

So for example, you'll find lots of Arabic words through vast stretches of Africa where the languages aren't Arabic because various forms of science and culture were borrowed, and so you see these borrowed words. That even works in modern times with languages like Chinese and Japanese. That they've been borrowing lots of words from English for various technological things. So cognates can often be quite helpful.

Well, what does that leave as this mysterious word for crok? Now we're left with no underlined words here. That doesn't mean we know the exact answer. One possibility is really we should take a pair of these words and translate them as a word in the other language. There are lots of cases in which you have something that can be expressed as two words in one language, and one word in the other language. It can even happen in the same language that you just have an alternative. So you can either say, "I went into the room," or you can say, "I entered the room." So "went into," and "entered" they're kind of two possible paraphrases. One is two words, one is one word. So we have to allow for those kind of many to one alignments in various ways.

But the other possibility is that this is just a word that isn't translated. And in the statistical MT literature, those referred to as zero fatality words, that the idea is this word just doesn't have a key. That's there nothing that it becomes in the other language. And so that's a possible scenario.

And so in this – this is the kind of process from which we can use parallel text to learn how to translate. And the cute thing about this example from me to close on, is well, actually this example is really Spanish/English except that Kevin Knight took all of the words and changed them into these sort of made-up Sentory Aucturan words, but you actually have exactly the same translation problem where what we want to translate is, "Clients do not sell pharmaceuticals in Europe." And this is our parallel corpus. And what you can see in the alignment is exactly the kind of things that we're talking about.

So here we get the word order mismatches. So in Spanish you have the adjective appearing after the noun. So you got these crossed alignments that we were seeing here.

And then here, this was the example with the cognates and the zero futilities, so here is the cognates, so I'm sure what it is. I presume it's some kind of drug, and so its similar in both languages. And well, what was our dual futility word? The zero futility word was do. So this is just the kind of thing that happens commonly with languages. That you get these function words. So in English you're putting in this axillary verb "do" to go along with the negation, "do not sell." Whereas in Spanish, there just is no equivalent. You just have to note no sell, and that's all there is. So these kind of function words often appear as zero futility words.

Okay, so that's where I'll stop for today, and then next time we'll first of all take a little detour into [inaudible] language modeling which is a kind of core technology that we'll need to do machine translation, and then on Wednesday get back to machine translation proper.

[End of Audio]

Duration: 75 minutes