UNIVERSITY OF LEEDS

Information Extraction: Extracting Named Entities and Relations from Text

Professor Eric Atwell: Hello. This is Eric Atwell, and I'm going to today talk about information extraction. That is, extracting named entities and relations from a text.

And this lecture is based on a presentation by Diana Maynard. She's a researcher at the Natural Language Processing Group at University of Sheffield in the UK. Sheffield's about half an hour to an hour's drive from Leeds, depending on traffic. And in case you Google Diana Maynard, you may come across her, as well as being a chief researcher, she's also involved in ballroom dancing. So, that's something else to look at.

OK. So, what we're going to talk about-- first of all, what is information extraction and some applications of information extraction. And then we're going to look at two different approaches to building information extraction systems. We've seen lots of machine learning examples in artificial intelligence. But another way of doing it is a knowledge engineering approach to the experts to devise some rules.

And we're going to look at how you go about developing our rule-based named entity recognition system, and as an example from Sheffield, the MUSE, MUlti-Source Entity recognition system, to finish off some ideas about ongoing research and information extraction. So, first of all, what is information extraction?

Well, you may remember, we had a lecture on named entity recognition, recognizing in a text that New York is a named entity, not just two words. It's information retrieval like Google search, and you give it some keywords or queries, and the information retrieval system pulls documents which seem to match your keywords from a large text collection. And you've probably used information retrieval in terms of searching the web via Google or Yahoo! or some other search engine.

So, a Google search is one example of information retrieval. And, technically, the thing that's pulled are still called documents in information retrieval theory, even though there could be web pages or something else. But you must then still have to read or analyse the documents to find the answer to your questions, to find what it is you actually want.

Information extraction goes a bit further in that given a document, or a set of documents, it pulls facts and structured information. Essentially, it pulls out the named entities and the relations between those named entities from the contents of a large text collection, such as a corpus or corpora. You then don't have to read all the documents. You just have to analyse the facts it has pulled out. And you also get a link back to the original documents, so you can go back and check that it got it right.

So, why would you want to do this? Well, with traditional information retrieval query engines, getting out the facts can be quite difficult. For example, if I wanted to find out where has the queen visited in the last year or which places on the East Coast of the United States have had cases of West Nile virus-- well, first of all, you've got to think what are the search terms to you to try to get the web

pages. Queen, visiting, 2021-- depending on what last year is. And we probably want the queen of England, Queen Elizabeth. And you have to be careful with Queen, because there are various other queens you might find out. So, getting the search terms is quite difficult.

Information extraction would hopefully return information in a structured way to get information like the queen of England as an entity. And other names for her might be Elizabeth or Queen Elizabeth. Information retrieval would simply return the documents containing the information, hopefully somewhere if you're lucky. So, for various types of search, information extraction can be a better alternative to this information retrieval. Information extraction returns knowledge in some way at a deeper level for information retrieval.

You can use information extraction to pull out the entities and relations and construct a database, or knowledge base if you want to call it that, which contains the information in the documents in a more structured way. And it can be linked back to the documents as an alternative search. So, you can still search the documents. You can find the entities. And if you're not sure you've got the right answer, you can go back to the documents to read them.

So, even if the results of information extraction aren't always accurate entirely, they can still be valuable. Because you still can check them in the original text. So, it's particularly useful for access to news, to identify in news stories the major events, the major players, the entities, and the relationships between them, for example, foreign affairs or business news. So, if you wanted to find out what companies buying which other company, you can find out entities like companies and relationships between them, such as buying.

It's also very useful for research in scientific areas, particularly in medicine, and pharmacology, and genomics, where in the subfield there are lots of very specialized terms for specialized sorts of entities and specialized sorts of relations. So, being able to extract names of particular pharmacological entities, these drugs and what they do to each other or what they do to diseases, can be useful.

Let's look at a few simple examples to get you an idea. So, one developed at Sheffield was HSE, the Health and Safety Information Extraction System. So, this aims to look at company reports about health and safety. So, you may not be aware but every company including Leeds University, which is a sort of company, has to report to a health and safety inspector about health and safety issues. And then every year the health and safety inspector may come around and ask to look at the documents, and ask questions like, how many members of staff have died or had accidents in the last year. Hopefully, at Leeds University, not many deaths and not so many accidents.

Or another question-- is there someone responsible for health and safety in the university, or what measures have been put into place to improve health and safety in the workplace. Those are the sorts of questions that an inspector might want to ask. And if you just have information retrieval-- well, if you haven't got any system at all, you have to read the books to find the answer. Or information retrieval wouldn't be particularly helpful because it might return which documents have information about this in it, but the documents could be quite large. You still have to read the documents.

On the other hand, what HSE does is it identifies sentences about health and safety issues or entities, and extracts them, and populates a database with the entities and the relationships between them, so that hopefully you can query the database or knowledge base rather than having to read

everything. OK. So, here's another example, KIM, the Kibbutz Information Management System. So, if you don't, know a kibbutz is a collective cooperative farm commune in Israel. And Israel is well known as a place where there may be terrorist attacks due to unfortunate circumstances.

So, you can ask the Kibbutz Information System queries, and the queries are like English sentences. So, what they've done is they've taken lots of newspaper stories and extracted anything to do with kibbutzes, and entities around kibbutzes and relationships to do with kibbutzes and put them into a database. And then, being a database, you can query it using SQL or some sort of structured query language.

Because most ordinary people don't really understand how to write SQL queries, there is an interface which is sort of English sentence-like, or sort of quasi-mathematical. You can search for patterns where x is a-- and then you can specify, for example, x is a person, which name is known or is unknown. And x, furthermore, is involved in y, where y is an event which name contains kibbutz attack. And y took place in z, where z is a country which name is exactly equal to Israel.

So, this is a roundabout way of saying, I want to find examples of a person x-- we don't know the name of the person. And this person was involved in an event named a kibbutz attack. And the kibbutz attack took place in a country called Israel. In other words, I want to find people-- I want to find persons involved in kibbutz attacks in Israel. OK? I want to find persons involved in kibbutz attacks in Israel.

It's a bit more complicated. So, the query language isn't part of the information extraction mechanism. It's just the interface to the information extraction database. But here's some examples of replies, and these are newspaper stories. For example, the top one is Israel Kibbutz Attack Suspect Arrested, where probably the suspect is a person who's likely to be involved in a kibbutz attack in Israel. So, that one's right. If you're not sure, you can click on the article and read it. So, as well as having the answer, you also have the evidence for the answer, which if you're not sure you can go and check the evidence.

OK. Here's another example. This was Alias-i-- I think it's a New York company-- developed something called the Threat Tracker at the time of the Americans invading Iraq. You might think a threat tracker is some sort of satellite-based tracking system, which can see things which are threats. Actually, this is just tracking threats within text. So, it's not using vision or image processing. It's text analysis.

In their case, the threat-- the entities are dangerous entities such as Huda Ammash. So, Huda Ammash was one of the most wanted terrorist suspects when they were attacking. So, the Americans invaded Iraq, and they gave every soldier a pack of cards. And on the back of each card was a photo of one of the terrorists they wanted to catch. That way they could recognize them.

OK. So, here we have underlined or enbolded in yellow the mentions of the entity Huda Ammash found by the threat tracker. So, we have Mrs. Anthrax. That was her name, because she was supposed to be developing anthrax, a deadly disease. And we have Mrs. Anthrax again, and then we have Huda Salih Mahdi Ammash, which is her full name. And she was dubbed Mrs. Anthrax.

And then later on it says Ammash. That's just her surname. And then later we have 5 of Hearts. 5 of Hearts was the card where her photo was on the back. 5 of Hearts was another sort of name for her.

And then, finally, we have it described Her as a weapons of mass destruction suspect. So, Her is also indicating it. These are different references to the same entity.

So, named entity recognition is identification of proper names and other references to entities in text and their classification into a set of categories of interest. So, not just that you found Huda Ammash, but also that she is a person.

Other entity classes might be organizations like companies or committees or whatever, locations such as cities or countries, dates and times, or other types as well depending on what sort of task you have. So, if you're doing medical text recognition you might have medicine entities or disease entities.

So, named entity recognition is important as a foundation to build more complex information extraction. If you want to find out relations between named entities then first of all, we have to identify the entities and then you can go on to find out the relationships between them. And this provides tracking in the text sense, ontologies, and scenarios.

So, you might have tracking co-reference. Dr. Anthrax, Huda, and she all refer or co-refer to the same entity. You might have ontology relations like persons, events, and countries are all types of entities within an ontology and they're related in various ways. And the relations could be also from a tech. Like Dr. Head is an entity and Shiny Rocket Corporation is an entity, and in the sentence, "Dr. Head became the new director of Shiny Rocket Corporation," then the relationship is director, that sort of thing.

So, how do you build systems for recognizing this? Well, on the rest of the AI/MLC you've come across machine learning quite a lot. Machine learning uses statistical language modelling or machine learning. You have to have-- you don't-- you have to have people who understand machine learning. They don't necessarily have to understand the domain. If you're doing machine learning from medical texts you don't have to be a medical doctor.

It does require a large amount of training data annotated with the targets. So, someone has to go through the text and label this is a person, this is a company and so on. And typically machine learning models, particularly deep learning models, require a lot of computing processor and memory. Furthermore, if you want to make some changes-- for example, you're not just interested in people and companies but you're also interested in universities or public organizations separate from companies, and that means reannotation of the entire training corpus. You have to go back and everything that was labelled a company maybe have to be relabelled as a university because you're separating out industry companies from non-commercial companies or something.

OK, so that's the pros and cons of machine learning. Well, we'll have some of the disadvantages of machine learning. You need a lot of compute and a lot of effort to collect the data and annotate the data. The knowledge engineering approach, this used to be much more popular before powerful machine learning was actually feasible and this is to come up with hand-crafted rule-based systems.

This doesn't just apply for information extraction but to other tasks. If you remember, part of speech tagging I mentioned that at Helsinki University you could spend three years on a PhD just developing a rule-based system for part of speech tagging the language of choice, the English, Finnish, Russian, or whatever language you happen to like.

And they didn't need to have a lot of data to do this. You have to be experienced in the topic area. You have to know something about that the task. It requires-- you have to be smart. It requires human intuition to figure it out. You only need to have a small amount of example training data to figure out possible rules for deciding this is a noun and this is a verb. You don't need thousands of nouns and thousands of verbs, just a few of them to see typical contexts.

And the same thing for information extraction. You may not require high-performance computing. It may work on your laptop. But development time can be time-consuming. And if you want to change-- for example, you don't just want to identify companies but you want to separate out industry companies from non-profit companies then that may mean you don't have to suddenly have to go back to the old data set but you do have to check all the rules you've got so far. And the rule set can be quite large.

OK, so that's a knowledge engineering approach, and I'm going to-- because you've had lots of examples of machine learning in other parts of the course I'm going to focus on an example which applies the knowledge engineering approach just to show how it's done.

So, let's say first of all you want to recognize the named entities. So, why is this difficult? Well, the real problem in all language analysis is ambiguity. First of all, for a particular named entity you can have different ways of referring to it. So, John Smith, Mr. Smith, and John all refer to the same entity or can refer to the same entity. So, that's many names for one entity.

The other sort of ambiguity is one name could be several different entities. For example, John Smith, I just said that was one person, but actually there may be several people called John Smith, or even worse, there may be a company called John Smith. If ever you come to Leeds and then you drive from Leeds to York you'll pass the John Smith Brewery where they make John Smith Beer. So, John Smith is the name of a company in Yorkshire.

OK, another example. June-- when I go to shopping at my local Lidl supermarket there's June behind the checkout, and that's different from June the month. Or Washington, another example. I'm told there's a famous American called Washington, I've not learned much about it-- Denzel Washington, I think-- whereas on the other hand, I do know Washington. I've visited. There's a town in the North of England near to Newcastle. So, there's-- it can either be a location or a person. 1945, that's ambiguous as that could be a date, the year that the Second World War ended, or it could be quarter to 8:00 in the evening, about time to have dinner.

So, ambiguity is very-- even common, ordinary words may be ambiguous. So, a word like may, because it's in lowercase we think it's just I may like you, but typically you can't be confident that capital letters are always going to be there so this could be the proper noun, May, which is the month May.

OK, that's just individual words, and within a document the document structure and style and even the particular domain of a document or the genre of a type of a document can give additional information. Punctuation, spelling, spacing, and formatting are all issues to take into account.

So, for example, if I come across a piece of text like Department of Computing and Maths, Manchester Metropolitan University, Manchester, United Kingdom, I as a human immediately recognize this as an address that you might get at the top of a letter. So, somehow the information

extraction system, the named entity system has to recognize that this is a single entity and it's hierarchical and it's the address of the computing department, which is within Manchester Metropolitan University, which is part of Manchester, which is part of the United Kingdom.

Or a second example, when we have, "Tell me more about Leonardo da Vinci," but there's a greater-than sign and I as a human know that that greater-than sign means it's probably part of an email, it's a repeat of a previous email. So, there's additional information from the punctuation or the layout which you would have to know about.

Having given an idea of some of the challenges, how do you do named entity recognition? Well, the simplest baseline approach is simply to have lists. The system recognizes entities stored in its lists and the lists are called gazetteers. This is from tradition. A gazetteer used to be a list of place names or person names.

This is nice and simple because you basically go through a text. For every word you look it up in the gazetteer. If you find it then it is a named entity. If you don't find it it's not a named entity. Nice and easy. And it's an easy-to-read target for a new topic, a new language, a new domain. You just have to create a new list.

The disadvantage? Well, there are several, one of which is that first of all, you have to collect the lists in the first place and maintain them. So, if you're doing pharmaceutical named entity recognition then you have to make sure your list of pharmaceutical entities is up-to-date. Also, it can't deal with this ambiguity problem that we saw. It can't work out that John Smith and John are the same entity, or it can't work out that John Smith could be a person, or it could be a company.

So, what can you do on top of just having these lists or gazetteers? The first thing you can do is to start looking at context. There may be internal evidence within the unnamed entity. If you come across, for example, a word starting with-- a capitalized word starting with a capital letter followed by City or Forest or Centre or River, for example, Sherwood Forest, then that is probably a named entity, and it's probably a geographical location.

It doesn't always work. For example, Nottingham Forest-- anybody coming to England might know this-- is the name of a football club. It's not-- there isn't actually any forest left near Nottingham. They've cut it all down. Although Sherwood Forest is very near to Nottingham.

Another sort of rule is a capital letter at the start of a word followed by Street or Boulevard or Avenue or Crescent or Road is probably a named entity which is a location. For example, Portobello Street is a famous street or road in London. Again, this doesn't always work. For example, it's Christmas now and I've got lots of Quality Street. Quality Street is not a location. Quality Street is a type of chocolate sweets. So, Quality Street is commonly eaten around Christmas. So, that works most of the time. There are exceptions where it doesn't work.

There's also-- ambiguity is again a problem. We saw particularly if you're looking for the capital letter at the start of a word, that is something that works reasonably well in English. Other languages like Arabic don't have capital letters, or Hindi doesn't have capital letters so that's not going to work.

And even for English it's not always the case that the word starting with a capital letter is significant as a name because we do have this rather awkward rule that the first word of every sentence has to

start with a capital letter. So, if a sentence starts "All-American bank," that could be a named entity, whereas if it starts "All state police" then the entity is actually "State Police," not "All-State Police."

Another problem is if you have sequences of these like John F. Kennedy, that could be the airport in-- New York? Yes, it is John F. Kennedy Airport I've been to, or it could be the name of a president in America. Or Philip Morris looks like a person's name but is also the name of a company.

You also have structural ambiguity that company names often include words like "and" and "of" in them. So, "Cable and Wireless" is quite a big company, whereas "Microsoft and Dell" is two companies, not one. Or a message from "City Hospital for John Smith," there's entities "City Hospital" and "John Smith," whereas you can have a "Center for Computational Linguistics," and that is one entity. So, that's complicated.

So, even if you have sequences of words, you then have to start looking at contexts beyond the sequences of words to help disambiguate. So, for example, "David Walton" and "Goldman Sachs" both look like names of people or names of companies, we're not quite sure. But if you come across "David Walton of Goldman Sachs" then you know that x of y implies x belongs to y and-- well, that's probably a person, David Walton, belonging to the company Goldman Sachs. So, we have person of organization. And you can then-- you can generate rules like this. These are semantic patterns.

And how do you do this? Well, you use a quick concordance. For example, if you remember in Sketch Engine you can look for a word like "David" and then look at words which appear around it to give you co-concordances or context around that, and if you find repeated contextual patterns then that gives you a clue to possible patterns. Hence, you've got out these patterns and you can search for more patterns including more of these things and just repeat over and over again.

So, these are the sort of patterns that you might extract. If you search using a concordance you might find a person followed by the word "owns" followed by another entity. It's likely that the first entity is a person and the second is some amount of money. Or "x joined y," x is likely to be a person and y is likely to be an organization and so on.

And this doesn't just find you the classes or types of entities, but it also tells you the relations between the entities. So, this finding-- if it's a pattern "person earns money," if you find "John Smith earns 15 pounds," then you've identified "John Smith" as a person as opposed to a company, and "15 pounds" as a money entity. And you've also found the relationship between these two is the earning relationship. So, these semantic patterns find the entity types and the relation types.

So, that's, in a nutshell, how it's done in the knowledge engineering or rule-based system approach. You can build similar things using machine learning but for machine learning you actually have to have hundreds of examples of each type. So, you have to spend a lot of time getting hold of a corpus and getting people to manually mark up the named entities, the types of the entities, and the relationship between the entities, and then you can try to learn from the corpus. So, you can imagine doing the markup up can take a lot longer than just building the rules.

Let's look at MUSE. This is an example of the multi-source entity recognition system built at Sheffield University using their GATE, General Architecture for Text Engineering. And to find out more about ways you can go to the GATE website, gate.ac.uk. Notice it's not gate.co.uk but gate.ac. The ac is Academic Community.

So, this is an academic research project, and it performs-- the MUSE system performs named entity recognition and code reference on different text types and different genres. It uses this knowledge engineering approach with hand-crafted rules, and it works just as well as machine learning-based tools.

And because it's hand-crafted one of the advantages is that they've been able to build rule-based systems for different modules separately. There's one to look at formatting. Is it a newspaper? Is it a letter or whatever? There's another for tokenization, just chopping it up into words. There's another for sentence-splitting, splitting it up into sentences, where the full stops and question marks are. Then you can have a part-of-speech tagger which labels the words as verb, adjective, noun, and separates out common nouns for proper nouns. Then you can have the gazetteer lookup which looks up each word in a dictionary to see if it's a named entity or not.

Then you have these sorts of semantic grammars, for example, if you've got "entity earns entity" and the first entity as a person and the second entity is amount of money. You can also do coreference, so if you've got "Mr. Smith" somewhere and later on you've got "John" then both these names refer to the same entity. You can also have pronoun coreference. If you've got "John" as an entity and then later on you've got "he" then "he" probably refers to the same entity as "John" and the pronoun "he" can be resolved.

The nice thing about this is you could adapt this fairly straightforwardly to other languages. First of all we did it for English because that's what they speak in Sheffield and then they developed it for other European languages like French and German and Romanian and Bulgarian and Russian. And then just as an intellectual exercise almost they tried it for some other languages just to see how difficult it is.

Cebuano is a native language from South America. Hindi is a native language in India with millions of speakers or hundreds of millions of speakers. Chinese, Arabic, and so on, and because of these different components you could separate out the language-dependent components from the language-independent components.

So, for example, coreference resolution works once you recognize what the entities are fairly well. So, for example, we mentioned Cebuano is not a language I know. It's used in South America by natives there, it uses the Latin or Roman alphabet the same as English or French or German, it has got capital letters denoting named-- proper names, sorry, names of things, so that works.

On the other hand, there's very few resources for this language. In South America the official languages tend to be Spanish and Portuguese so most of the language resources are for Spanish and Portuguese and not for these lesser-known Indian native languages. That makes it medium difficulty.

Hindi was another example they tried to develop for. This has a completely different alphabet, doesn't use the same script at all. So, you'd have to have different character encodings, it doesn't have capital letters, words do have spaces between them so you can work out what the words are at least. And because Hindi is one of the major languages of India and India is a big country with lots of computing research going on there are lots of resources available for it. That makes it medium difficulty for different reasons.

So, if you're going to adapt MUSE to other languages, what do you need? Well, you need to have a lot of support for other scripts, other character sets, and that could take up two-thirds of the actual time effort of development. And it's fairly-- if you like, it's not particularly AI research dealing with different character strings, character sets, but it is important and time-consuming.

You also need to get hold of dictionaries for mapping. Ideally it-- very often you can get hold of an English to the other language dictionary because English is everywhere. So, for all-- if not for Cebuano you could get hold of a Spanish-to-Cebuano dictionary because-- or Portuguese-to-Cebuano dictionary. So, typically for most languages there is a language dictionary and a bilingual dictionary comparing that language to one of the major European languages.

You also need to get hold of a corpus for evaluation. So, even though you don't need it for training machine learning you do have at least a small corpus where someone has manually annotated the names of the entities and the types of the entities and the relations so that you can check. You can then run your MUSE over this and come up with some results and then compare the results against the actual truth, what the human annotations were. So, you do still need to annotate at least a small test corpus for evaluation.

You also have to be quite clever in looking for the internet to find resources. If you want to have a list of names of people, names of places, names of companies then maybe you can find these on the internet. There may be phone books or Yellow Pages or other resources on the web that you can mine. And then you can do text extraction from that to try to get the information you want.

As I said, a big part of the problem is dealing with different character sets. So, they actually developed a GATE Unicode kit on top of just the core Java resources which allows you to type in and have virtual keyboards for lots of different languages and display lots of different languages. So, that's the input, and the output allows you to display, for example, this is Chinese and in green we have a location and in blue we have some persons and in purple we have the organization. I don't speak Chinese, but I have shown this to a Chinese speaker, and they have verified that they've got this right. So, this is just an example.

To do proper evaluation, as we saw with Word2vec, the Mikolov research team, as well as the developing Word2vec they also developed a very large evaluation test set. So, to test or evaluate information extraction you can just show some examples. What we should really do is come up with a common, very large, standard data test set which other people can use, too.

So, what's coming up in information extraction? Well, as I said, one very important thing is extraction evaluation sets. So, there's ongoing research into developing large corpora with information extraction labelled. You also need to develop tools for the semantic web, supposedly still coming, as you should be able to search for meaning on the web pages rather than just words on web pages.

There's definitely a need for information extraction in bioinformatics and medicine. Particularly with COVID there's been a lot of funding for information extraction to try to find information about COVID, COVID sources, COVID cures, and so on. Another big area of information extraction is in finance industry. For example, to predict share prices you probably want to see what is happening in the world, and news can directly or indirectly affect costs of commodities, which then affect share prices.

You do want to be able to not just extract information, but answer questions, and to answer questions what Google has to do is find the web pages and then extract entities from that to answer the question. So, there's cross-fertilization between information extraction and information retrieval.

Another thing apart from finding the entities is you need to find the relations between the entities. I haven't really said much about this in this lecture, but your Jurafsky and Martin textbook has a chapter on information extraction where they talk a lot more about the algorithms for machine learning, that type of information extraction. And also, the algorithms for relation extraction. So, read the chapter to find out more about this stuff.

OK, so that's it for now. We've looked at information extraction, which is extracting named entities and relations from text rather than just information retrieval, which is finding the documents. Once you've found the documents the information extraction can find useful things in the documents.

We looked at a few applications like the threat tracker or the kibbutz information system or the MUSE system. We saw that apart from machine learning, another way of building AI systems is by knowledge engineering, by knowledge engineers building a rule-based system. We saw rule-based named entity recognition is possible by coming up with rules at different levels, rules for named entities, rules for labelling the entities.

And the MUSE system, the MUlti-Source Entity Recognition System, is one such example developed at Sheffield University by the Natural Language Processing Research Group there. And we finally looked at some areas for ongoing research and information extraction. And if you read the textbook in Jurafsky and Martin textbook chapter, you'll find more information about ongoing research, including machine learning research to do information extraction.

OK, hopefully now you've got an overview. Please go and read the chapter in the book for a lot more detail of the algorithms and the ongoing research. Bye for now and I'll stop there.