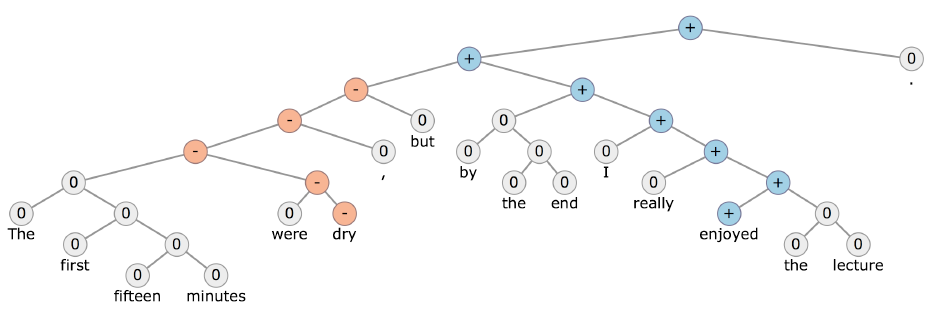
**Intro to NLP and Deep Learning**

Stanford University

CS224d: Deep Learning for NLP

Richard Socher

**Welcome**



**What is Natural Language Processing (NLP)?**

Natural language processing is a field at the intersection of computer science, artificial intelligence, and linguistics. Our goal is for computers to process or “understand” natural language in order to perform tasks that are useful, e.g. Question Answering.

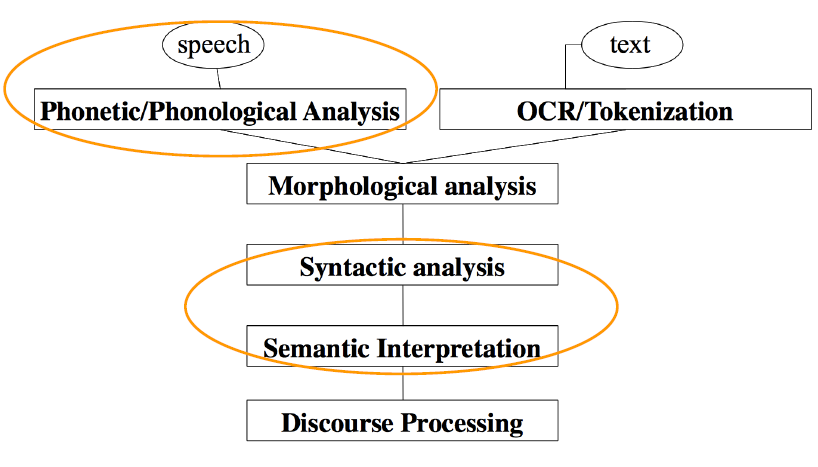
Fully understanding and representing the whole meaning of language (or even defining it) is an elusive goal. We won’t to go in details. But, we want to be realistic in our expectation. In the end, we usually have a very concrete task that we think it’s useful to solve.

When we say meaning, understanding, and reasoning, and so on, we need very complex function that we learn to represent them and to predict or solve the useful task.

Perfect language understanding is AI-complete. It means link to thought. Actually, we don’t have a way of representing all kinds of thought in a computer.

**NLP Levels**

There are different levels in Natural Language Processing.



Syntactic analysis is trying to understand how words are put together the form meaning of whole sentences. It’s like a grammar, syntax, structure of the language.

Traditionally, once we have syntactic analysis, we’ll then go to understand what the sentence actually means. Eventually, we might want to use the sentence meaning to have a full this course.

Major focus will be on syntactic and semantic interpretation.

Notice that you might actually go directly from list of a word to what does that list of a word actually means. That is, we may skip some of these levels, that have been traditionally thought to be required full understanding language and go directly to semantic interpretation because that means the meaning is at the end when we communicate (it’s less about the structure that we use) 즉, word embedding 표현법을 사용하면 전통적인 NLP단계를 거치지 않아도 곧바로 semantic analysis를 할 수 있다.

**(A tiny sample of) NLP Applications**

Applications range from simple to complex:

- Spell checking, keyword search, finding synonyms

- Extracting specific types of information from websites such as

: product price, dates, location, people or company names

- Classifying, reading level of school texts, positive/negative sentiment of longer documents

- Machine translation, which really captures a lot of the complexity of language

- Spoken dialog systems

- Complex question answering

**NLP in Industry**

• Search (written and spoken)

• Online advertisement

• Automated/assisted translation

• Sentiment analysis for marketing or finance/trading

* It’s very interesting task because it’s one of those general classifiers. Once trained a really good accurate sentiment classifier, you can use it across a whole host of different task. (e.g., understand movie reviews, customer sentiment)
* 엔헤서웨이가 출연한 영화가 흥행에 성공하면, 헤서웨이 회사가 언급이 많이된다. (search a lot) -> entity disambiguation property를 사용하지 않았다. 헤서웨이가 사람이름인지 회사이름인지 구분할 필요성이 있다.

• Speech recognition

• Automating customer support

**Why is NLP hard?**

There’s a lot of complexity in representing, learning and using linguistic/situational/world/visual knowledge.

• “Jane hit June and then **she** [fell/ran].”

* she가 지칭하는 이름은 무엇인가? It depending on the meaning of the verb that follows it
* 전통적으로 NLP를 처음 시작할 때, 단어들을 단지 List up을 했다. 예외가 많다. 단순히 규칙으로만 모두 정의하기 쉽지 않다.

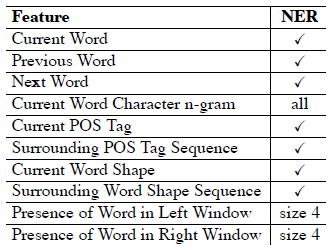
• **Ambiguity** : “I made her duck”

* There are a couple of different interpretations of that sentence.

**What’s Deep Learning (DL)?**

Deep learning is a subfield of machine learning. Basically, deep learning tackles following problems that most machine learning methods work really well because it turned out that there is human learning (human-designed representations) and input feature. For example, features for finding name entities like locations or organization names.

Give all these kinds of features to machine learning algorithm like logistic regression or conditional random field, to make a very accurate predictions.



* Features like is the first letter capitalized, is the every letter capitalized, is the no letter capitalized
* POS tags: is this a noun, verb, and adjective, …..
* is the word is in front it

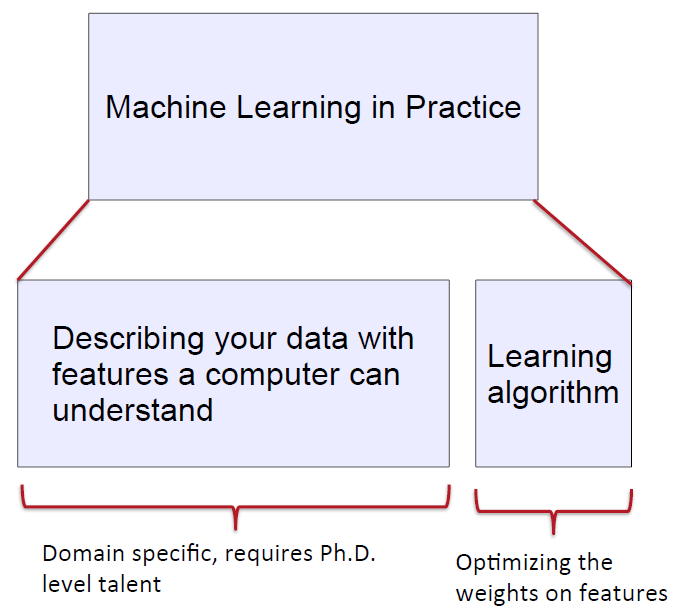
A lot of different kinds of features were given to the algorithm and only then was able to find out that “this is a location”, “this is a city name” or something like that. 예를 들어, 장소 단어를 분류하는 문제일 때, 먼저 장소 단어들을 list up을 하고 list up된 단어들을 보고 일련의 패턴들을 찾는다. 그리고 해당 패턴(e.g., 첫 글자는 대문자)들을 feature로 사용된다.

Now, that’s not very attractive if you try to build an algorithm and you had to actually manually sit down and look at all the data and say “Well still making this mistake, maybe I should add this kind of features to solve that specific problem because that way will be very dependent on the kind of texted you’re looking at as a human. In many cases, it will actually be too specific to that dataset that you’re looking at and then if you have another language or another kind of dataset, you have to look again and manually look at over all these features.

The beauty of deep learning will be that we will actually try to automate that process.

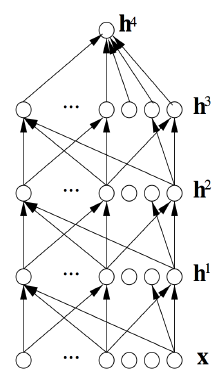
**Machine Learning vs Deep Learning**

1. Machine Learning



* You spend the majority of time(90%~95%) to describe your data with features a computer can understand
* Optimizing function is very simple function (10%~5%)

1. Deep Learning (DL)



* DL basically try to learning these representations and you can see deep learning as a type of representation learning
* Representation learning attempts to automatically learn good features or representations by giving it the raw data **x** (e.g., in image raw pixels, raw words, or just raw characters)
* <raw characters, positive or negative sentence> 이러한 training example set를 모델에 주고 학습을 시킨다. 여기서 DL은 단어간의 상관 관계나 단어 또는 character간의 순서 등을 manually하게 feature로 정할 필요가 없다. 그냥 단지 label이 달린 training example만 쏟아주면 된다.
* Deep learning algorithms attempt to learn (multiple levels of) representation and an output
* Word2Vec models are actually not Deep models, but shallow models (하지만, 가지고 있는 철학은 비슷하다.)
* We will actually give it the raw words, raw speech signals, or raw pixels or a combination of all these things and it will learn how to combine them and how to combine the features of those features various different deep levels, so in the end, we completely automated
* 그럼 feature들이 잘 learning되었다는 것을 어떻게 아나? → Many times, we can actually project them down into a two-dimensional or three-dimensional space that we can visualize and then we’ll actually see very interesting patterns that emerged from these models in often unsupervised way. So, there will be some ways we can visualize it but also almost part of the art. Sometimes, you get to know the model really well and you get to tune it to work pretty well. (visualization 또는 interpretation하는 방법들이 많이 연구되고 있다.)

**On the history and term of “Deep Learning”**

Deep learning actually also includes a lot of so-called graphical models and probabilistic generative models but we’ll not focus on those, we will focus the entirely of the class on neural network family, which is really now the dominant model family inside the field of deep learning research as well as industry.

You can think that this whole deep learning thing and neural networks isn’t just kind of clever terminology for combining a bunch of logistic regression units. One thing you notice is we will have a lot of logistic regressions, kind of put together in all kinds of different ways. At the same time, there are a lot of interesting modeling principles such as ‘end-to-end’ learning. (I give you the raw input, and I give you the final output, you learn from the final output all the way to the raw input and everything in between. Really putting the artificial back into artificial intelligence)

• Only clever terminology for stacked logistic regression units?

* Somewhat, but interesting modeling principles (end-to-end) and actual connections to neuroscience in some cases.

• We will not take a historical approach but instead focus on methods which work well on NLP problems now

• For history of deep learning models (starting~1960s), see: Deep Learning in Neural Networks: An Overview by Schmidhuber

**Reasons for Exploring Deep Learning**

• Manually designed features are often over-specified, incomplete and take a long time to design and validate those kinds of features

• Learned Features are easy to adapt, fast to learn automatically (10년전에는 엄청 오래 걸리던 일을 딥러닝을 사용하면 손쉽고 빠르게 처리할 수 있다.)

• Deep learning provides a very flexible, (almost?) universal, learnable framework for representing world, visual and linguistic information.

• Deep learning can learn unsupervised (just figure out interesting relationships and semantic meaning of words just from looking at a lot of raw text data) and supervised (with specific labels like positive/negative, trying to very accurately predict a certain outcome)

• In 2006, deep learning techniques started outperforming other machine learning techniques. Why now?

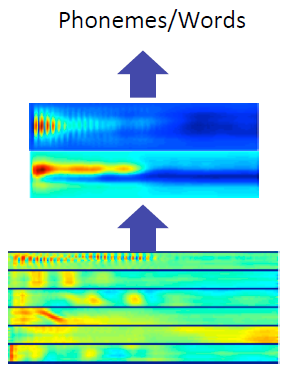
* Deep learning techniques benefit a lot more from the data because we will ask the models to learn the features by itself, it also requires a lot of data. When you give it to millions of examples, then it can start to figure out all these patterns by itself without you hand-holding and describing various features.
* Faster machines and multicore CPU/GPU help DL
* New models, algorithms, ideas

→ Improved performance (first in speech and vision, then NLP)

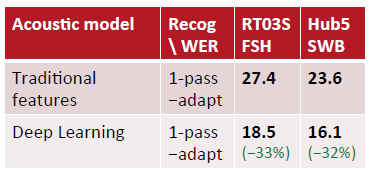
**Deep Learning for Speech**

• The first breakthrough results of “deep learning” on large datasets happened in speech recognition

• Context-Dependent Pre-trained Deep Neural Networks for Large Vocabulary Speech Recognition Dahl et al. (2010)



Deep learning started with speech and computer vision, but now has arrived natural language processing as well. Deep learning for speech was able to take in not quite just raw frequency, but some very raw features of the sound and then basically predict phonemes and words.



Here we see WER(Word Error Rate) dropped significantly.

**Deep Learning for Computer Vision**

• Most deep learning groups have (until 2 years ago) focused on computer vision

• Break through paper: ImageNet Classification with Deep Convolutional Neural Networks by Krizhevsky et al. 2012

We understand neural network models a lot better for images, so we know for instance when you take into raw pixels, the first layer of neural network basically finds edges and then once in the second layer we will combine some these edges into more complex textures and shapes and then in the layer 3, you’ll have even more complex kinds of combinations and ‘tire’ things like that.



In natural language processing, we won’t have beautiful visualization like that unfortunately and there won’t be as kind of obvious connections of “we put these two characters together and then they form kind of this positive meaning or something like that, but we will have some visualizations also that help us understand. (2D projection을 통해서)

Actually, in text it will be very hard to understand exactly what one word would be represented as for instance the example of ‘dock’. When we try to represent ‘dock’, it should be closed in our representation to a lot of verbs like hiding and things like that or should it be closer to other bird names.

**Deep Learning + NLP = Deep NLP**

• Combine ideas and goals of NLP and use representation learning (=deep learning) methods to solve them

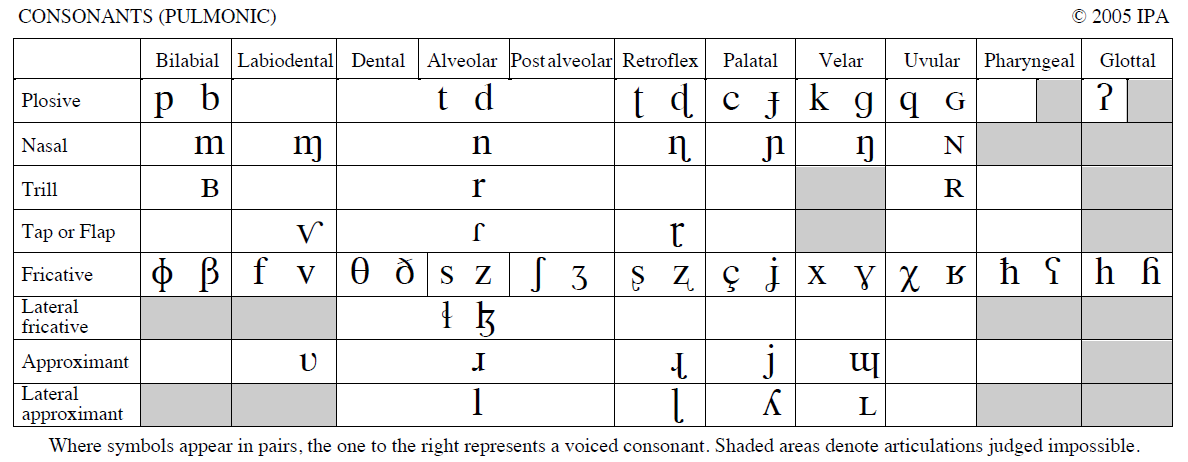
• Several big improvements in recent years across different NLP

- levels: speech, morphology, syntax, semantics

- applications: machine translation, sentiment analysis and question answering

**Representations at NLP Levels: Phonology (최소 소리단위)**

Originally if you try to understand words from spoken language, you need to transcribe it into written language and you study linguistics you would basically take a whole quarter on phonology like below picture.



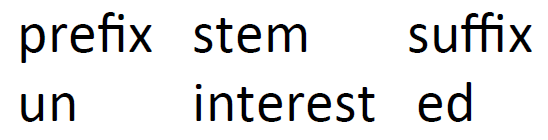
You basically spend a lot of time trying to describe all the subtleties of how human sound can be transcribed (문제점) and then basically try to automatically predict one of these discrete categories.

Now, in deep learning what we’ll do instead is just to describe it as a vector or a list of numbers and we won’t to try to make any of these fine distinctions between different ways to pronounce a word.

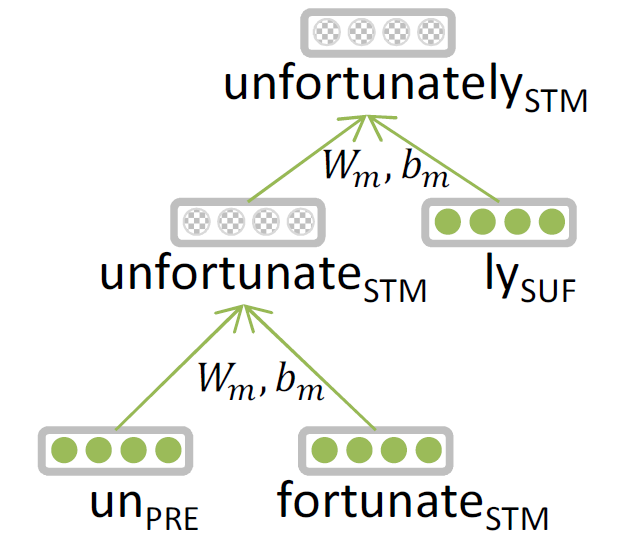
• Deep learning trains to predict phonemes (or words directly) from sound features and represent them as vectors

**Representations at NLP Levels: Morphology (형태소: 뜻을 가지는 최소 언어단위)**

Now moving on from just knowing what the sounds are, the next level might be understanding Morphemes, which is basically the field of morphology (we might want to understand a word like ‘uninterested’ might be chopped up into a prefix on and you see that prefix modifying lots of different words and changing potentially the meaning later on.



In deep learning, every morpheme is just a vector, a list of numbers and we will train a neural network just to combine two vectors into one vector and then we hope that the resulting vector captures whatever we need to capture for our task



PhD students at the Stanford NLP Group, they had to use a so-called recursive neural network. It’s also a family networks. That basically combines a list of numbers that represent ‘un’ and one that represent ‘fortunate’ and we will combine those two to ‘unfortunate’.

Notice that we won’t make any kind of explicit assumptions about what ‘unfortunate’ should look like. Instead, we just have to model to figure that out.

**Neural word vectors – visualization**

After we understand some of the parts that words are combined from, we want to understand the words themselves. Basically, unsupervised deep learning techniques will understand that certain words for similar to one another in certain relationships.

(Word vector를 만들기 위해서) We will learn just by giving it all of the texts of Wikipedia, neural network or even more shallow very simple techniques to map words into a vector space. This is really going to be one of the fundamental ideas of this class.

As long as we can map any kind of input into a list of numbers, we’re going to be in the world of deep learning and we can use all these different techniques (단어를 a vector로 표현하면 deep learning 기법을 사용할 수 있다). Again a word will be just a list of numbers and usually we will represent a word in 50 or 500 dimensions. Of course, we can visualize those if you want to understand what’s actually going on inside the model, by projecting down to two dimensions. Then, we actually see interesting patterns like below.



Companies like France and Germany are close to one another in some fuzzy way graphically. These are things that will be learned automatically from just giving it a bunch of raw text. However, it doesn’t always make sense. There will be problems when you represent everything as a single list of numbers but it will be very useful to solve a lot of different task.



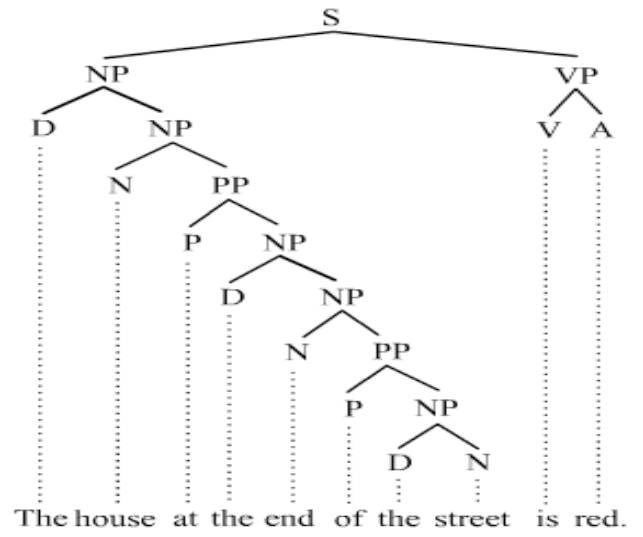
Here is a couple of verbs.

**Representations at NLP Levels: Syntax**

Now, we understand how to represent words, but word’s isolations are not that useful. Usually when we communicate, we combine words into longer sentences (useful structure).

• Traditional: Phrases Discrete categories like NP, VP

Traditionally, people would describe sentences were very discrete structures or discrete tree structures like below



You might say ‘the’ is a determiner, ‘street’ is a noun and ‘the street’ is discrete noun phrase. Unfortunately, just saying this is a noun phrase, it’s not that useful because you still don’t know what it actually means ‘the street’ (How does ‘the street’ relate to ‘the road’ or ‘the site alley’ or ‘the alleyway, which I was born’). It’s very hard to capture all of that in one discrete category but that has been to have traditionally the way to describe syntactic categories (just to structure of the grammar of language). In the end, you might say that whole thing is a sentence and the sentence is just the discrete category S.

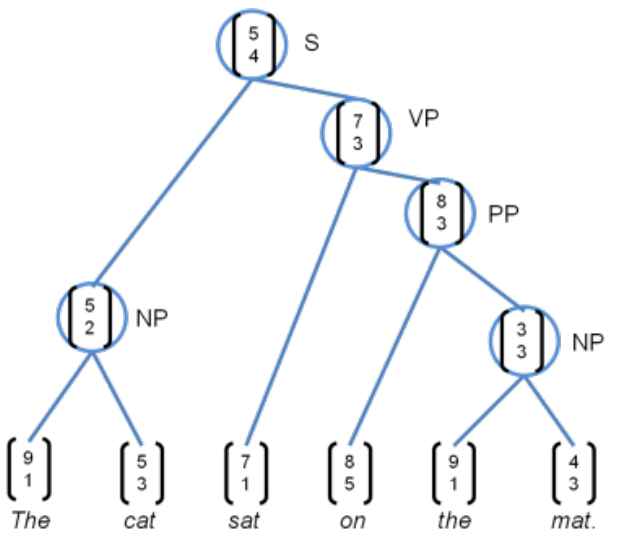
• Deep Learning :

- Every word and every phrase is a vector

- a neural network combines two vectors into one vector

- Socher et al. 2011

Now, in deep learning you start to see the pattern here since we already described all the words as vectors, we will also describe all the phrases as the vectors. And then we can actually compare the phrase ‘the cat’ with the phrase ‘the dog’. Now they’re not just noun phrase, but will actually have a vector and we hope that the vectors for phrases that are similar also similar to one another in some way (‘the cat’과 ‘the dog’가 vector상으로는 비슷하길 희망한다).



Since every word and phrase will be a vector, we can use standard neural networks that functions that vectors to other vectors to learn to make certain kinds of predictions.

**Representations at NLP Levels: Semantics**

Now, let’s go to the next level above which is semantics.

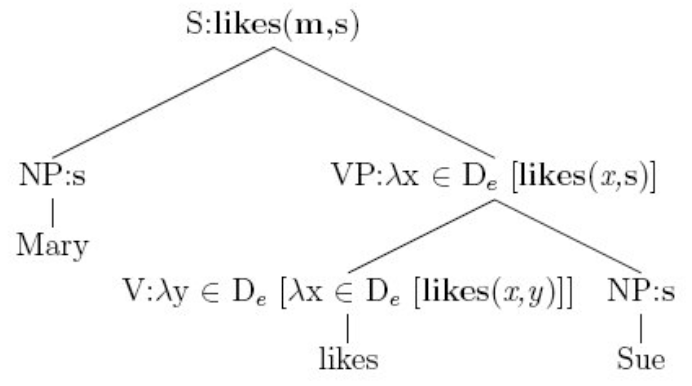
• Traditional: Lambda calculus

- Carefully engineered functions

- Take as inputs specific other functions

- No notion of similarity or fuzziness of language

We’re going through a long history here of the whole field of linguistics (morphology, phonology, syntax, semantic). You can literally have one to four quarters of classes about each of those separately. So, we’re not doing them complete and they’re actually very interesting but if you are trying to make predictions, trying to have a computer automatically understand and make predictions based on the inputs, it turns out we can in some sense get away with not having to understand all these subtle details. We can just make the computer do accurate predictions without us understanding all the subtle notions of these different layers of representation.



Traditionally, if we try to understand “Mary likes Sue” sentence, you would go to eventually what we call lambda calculus where ‘likes’ is this complex function likes(x, y), that takes as input some other variable y and then takes as input another variable x and then can basically combine these different representations.

‘Sue’ is just represented as some discrete symbol s and when we combine ‘sue’ with the verb ‘likes’ then what will get is likes(x, s). 그 다음은 likes(m, s) 이런 식으로 merge되어 간다.

If we try to ask how similar is that sentence ‘Mary likes Sue’ to the sentence ‘Mary and sue enjoy hanging out together’, that sentence already will be quite tough for lambda calculus to represent it. And they’re very discrete and they’re not too fuzzy long list of numbers and they were not in a vector space and so there’s no notion of similarity or the fuzziness of language when we use these kinds of representations.

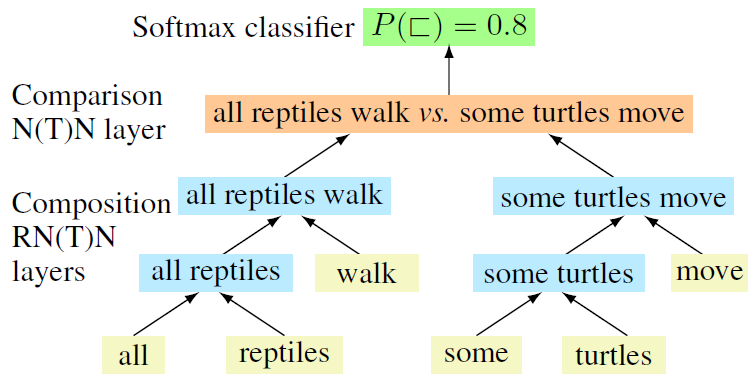
• Deep Learning:

- Every word and every phrase and even every logical expression is a vector

- a neural network combines two vectors into one vector

- Bowman et al. 2014

Now, in deep learning every word and phrase and even logical expression will just be a vector. Because of that we can combine different vectors using neural networks into other vectors and then we can still it turns out we get a classification problem. For instance, in the sentences like “all reptiles walk” vs “some turtles move”, we can ask does one actually locally entail the other.



Instead of having to define turtles are subsets of reptiles and moving is general expression and walking is a special subtype of moving, we consider that everything is a vector.

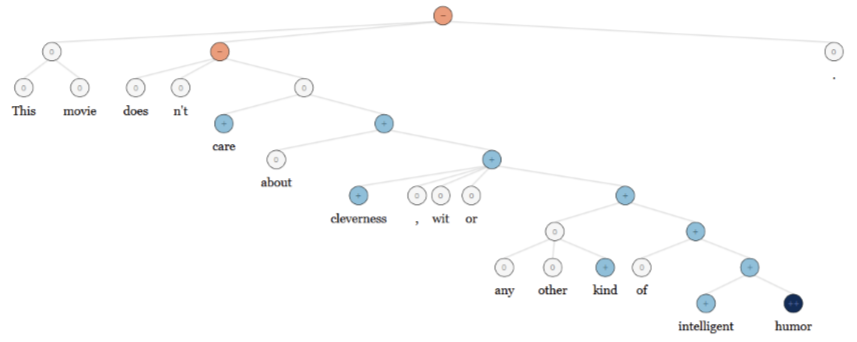
Let me just training model with the final output that I care about with a simple logistic regression at the top and just say yes or no. This two sentences are in the relationship or they’re not. And I learned everything in between and eventually amazingly the model without us having to manually define that reptiles as a super group of turtles, can figure it out to make the final correct prediction.

정리

We basically were able to see that for all these different levels of linguistic representation starting from speech to phonemes and morphemes, words, phrases, grammatical structure and the meaning, everything will essentially be mapped to vector.

**NLP Applications: Sentiment Analysis**

In traditional approach, you have to have curated sentiment dictionaries (all the positive words: wonderful, awesome, amazing,… and all the negative words: bad, …) combined with either bag-of-words representations (ignoring word order) or hand-designed negation features (ain’t gonna capture everything) 하지만 not good, not awesome 등 모든 표현들을 사전에 manually하게 구축하기란 쉽지 않다. not awesome은 중립적인 표현이 될 수도 있는 것처럼 사람이 규칙을 정하기엔 정확하지도 않다.



- care +와 +가 연결되는 상위 node가 0 (중립)이 된 것은 misclassification이다.

• Same deep learning model that was used for morphology, syntax and logical semantics can be used → Recursive Neural Network

Now, in deep learning, what we’re able to do is we basically give it a lot of examples of phrases and sentences and we can train a single model that takes in these raw labeled phrases and makes a final prediction and then was able to outperform all this human engineering of this domain of sentiment.

(딥러닝에서는) 위의 문장에서 positive단어 많았음에도 불구하고 does not의 단어 때문에 negative라고 classification할 수 있었다.

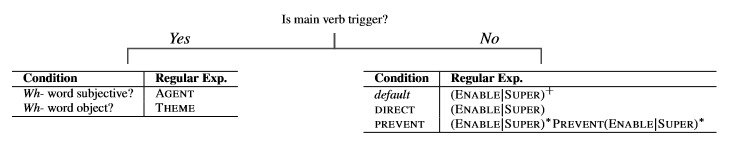
Again, we didn’t have to give the algorithm this kind of rule like not caring about positive things and so we just learned it automatically.

**Question Answering**

This is one of really tough problem in NLP where generally you need a lot of feature engineering.

• Common: A lot of feature engineering to capture world and other knowledge, e.g. regular expressions, Berant et al. (2014)

You may have regular expressions and find specific features is like WH question for subject or object and you define whose the agent of that sentence and things like that. So, pretty complex kinds of features that you have to engineer to do well.



• DL: Same deep learning model that was used for morphology, syntax, logical semantics and sentiment can be used.

Basically in deep learning there some folks that used the exact same model architecture or the same kind of neural network architecture where we use morphology, syntax, logical semantics and sentiment. In this case, it was a recursive neural network that used for all these different levels. It turns out to be also able to answer questions for more complex kinds of things. In the end, it stored a lot of facts that were asked about in training in vectors

When you zoom into this visualization, the model had automatically learned like this:



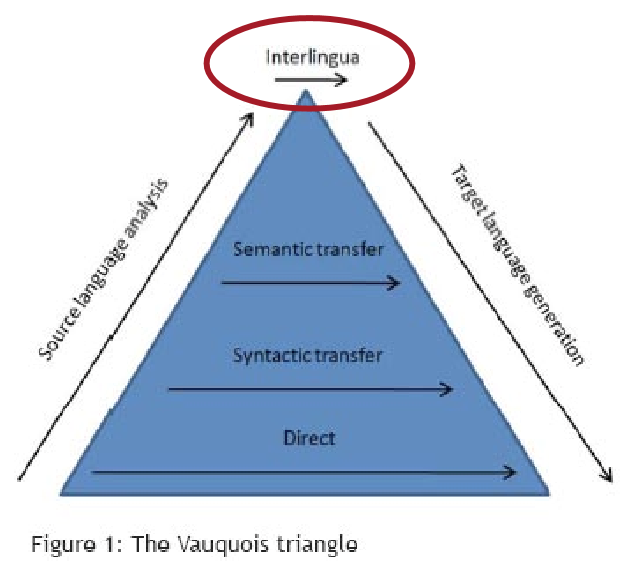
We didn’t teach about what presidents are and what different times. (서로 다른 entity들이 Vector space상에 grouping되어있다.) The model learned that automatically just by having asked and seen a lot of answers for different kinds of questions.

**Machine Translation**

One of the toughest tasks in NLP. There are a lot of different techniques that people have tried to translate from one language into another language. 예전의 방법론들: ① Some people said well first we need to maybe we can directly translate from one sentence to the other. ② But other people said no we first need to understand grammatical structure and then one grammatical structure can be translated to another grammatical structure. ③ Another said no what we really need is to understand meaning first and then we can translate easily from one meaning into another language that same sentence.

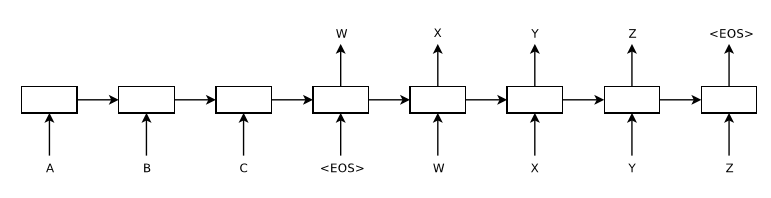
So, there traditionally extremely complex kinds of models that require hundreds of gigabytes of RAM if you want to keep because we have to store all possible phrase pairs and things like that.

What do you think is the interlingua for the DL approach to tranlation?



Vectors!!! So, everything will be mapped into vector.

In the below model, you basically take any kind of input language and then you translate from that vector you generate translation sentence.



• Source sentence mapped to vector, then output sentence generated.

• Sequence to Sequence Learning with Neural Networks by Sutskever et al. 2014; Luong el al. 2016

• About to replace very complex hand engineered architectures