**Logic in nlp**

Allow us to define entailment ie if one thing is true then the other must be tru (it entails it)

Allow us to be more precise when it comes to this reasoning

Firtorder logic: lets us expressing relationships between en**ities**

Maps things to true or false

Arg = lady gag is sing? Can be evaluated to tru or false

Sings is filter through a data base

Quantification is a feature of first order logic that is very helpful



Everyone who sings or dances preforms can be repped as

Or there exists operator

Can be a source of ambiguity

These logical reps are hard to work with from an NLP perspective

We can use lambda calculus reps

Allows us to build bigger pieces from larger [pieces

Good for reasoning and information extraction

Only makes sense in certain domains

**Montague Semantics**

More concreate and teid to consituancy parsing

“lady gaga sings”

Expresses something about the world that is either true or false

Denotation evaluate an expression again a database

Denotation of this string is an entity (like an id that works as a pointer)

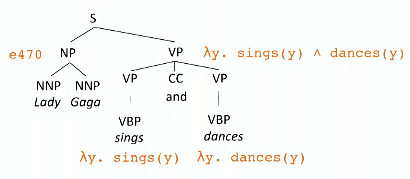
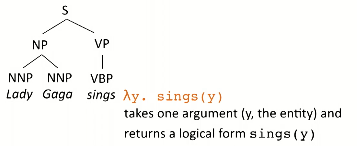
Sings(id) = tru or false

Lady gag id = e470

So how to go from statement to a more formal representations

Build up representation in a composistional way

Must tak and interpret with respect to our world



This a lambda function with one argument

The entity is the noun and the y is from the verb phrase

Sings(e470)

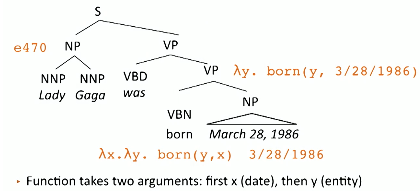
So syntaxic parsing helps us build up this sematic information

You can combine arguments as shown

General rules:

 how to join verb phrases

how to deal with sentence of verb phrase and noun phrase

What if you have more than one noun in your sentence

Verb combines with object and then it combines with a subject

Final expression needs both arguments

How to handle past tense

There are all sorts of trick thing in language

Adverbs, manner, can treat like optional arguments

Now it is hard to see how they connect

Some expressions will stress the frame work

Cats eat mice -> what does this actually mean in a concrete mathematical way

This had led people to use neural nets so that we don’t have to actually think about it

Good for data base qureyinf

**CCG**

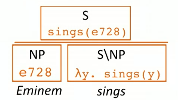
Sematic parsing formalism known as combinatorial categorical grammar

Formal bridge between syntax and semantics

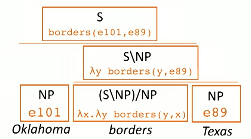
Instead of verb phrase has S\NP -> if combie with np on left forms a sentence

This is how to think about verbs

So when we build this tree we do correct syntax



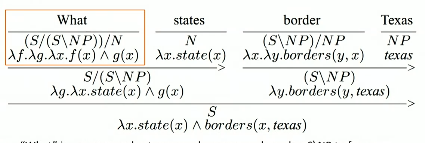
Apply the lambda calc to Eminem



Gor more complex sentences we need to have an np both on the left and on the right

If have a sentence like “what states boarder texas”

Can have bare nouns like states in thuns sentence



Shows what what neeed to make a full and happy sentence

Trick to put together – but canbe backsolved

Many entries in first level lexicon

How to pick correct entry from the lexicon

Need super taggers

P[arsing is much easier once you have the correct tags

Problem is we don’t know the derivation

We have to infere without really knowing

So any mapping between these thing is valid

Only supervison is local forma and correct ansers

Parsers are hard to build

So instead we use sequence to sequence model them we can map from sequence to lambda

Let neural net figure it out for us

No as simple as appears

Is more flexible and we have gotten better at building them for sure

**Seq2Seq**

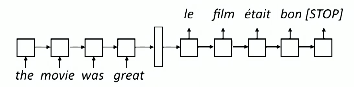
Allow us to flexibly hand mapping one sequence to another sequence

So map from text into lambda representations

Can create a parse tree

Super flexible framework that address a lot of NLP problems

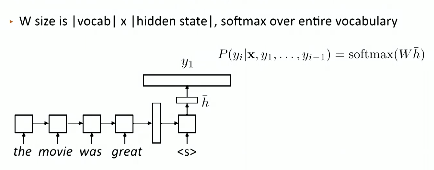
Encode a sequence into a fixed length sentence



So like one per cell

So at each time step generate the next word conditioned on prevous word and hidden state

W size is vocab x hidden state to g



Gives a probability distribution of word based on previous word

Have a decoder with seprate paramters from encoder

Will become a language. Model – can learn reasonable decoder output

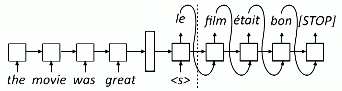
As go to subsequent iterations

Need to take decoder at time 1 and find the argmax

No fundamental need to do this

Need to take computation graph upto point to feed into next time step

Keep going until decoder produces the stop symbol



Condition on input and what has come before in out put

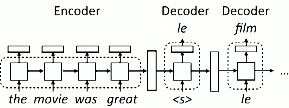
Flexible framework

**Seq2Seq training and implementation**

Unlike RNN seq2seq need to be evauate by decoder at each time step

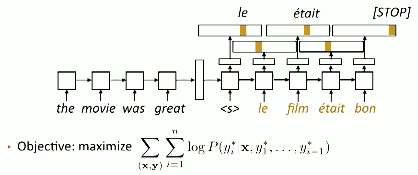
Theoretically same but lm does fix sequ

This is more generative as an example



Think od decoder as a separate mdolule that evalute at each step

Need to maximize the likelihood of the observed data

Can compute the log prob

One term in loss for each word in a target sentence

Train through teacher forcing

Assume matches gold and predict the next word

Strange that we must assume we did evrythign correctly

Can go off the rails

So model needs to act right even with its own interp

Begin with gold and the decay until it uses it’s own predictions

In order to solve need reienforcement learning - need to focus on max rewards

Implementation:

Pad input and out put for decoding/ encoding

The encorder is an RNN module

Decoder excute one step at time

Batching is pretty difficult with this nedd to run across multiple time steps

Bea search – want to do is find the most likely out put sequence if you greedily max won’t get highest probability sequence

Get distribution over probable token and this populates our beam

Each translation has a particular score

Le flim vs la flim should

vectors are important to keep around

the encode maps sentences to a vector

the decoder takes a hidden stat and an out pu prefixe and and produces a new hidden state and out put feature

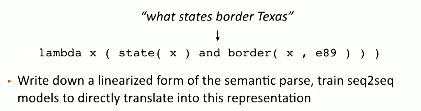
a wide variety of models can be used

transformers are a very useful sitch

rnns and seq2seq can be used but are not always together (de coupled

**Seq2Seq Semantic Parsing**

What states boarder text can be written as sematic parsing



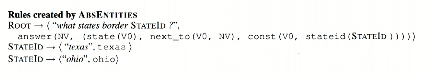
Benefit no grammar engineering

But might producing something illogical - not correct # of closed paraenthese

Want to encode same typ of variances

Ohio can sub for Texas -way to create variance

Wanted to have data augmentation – can do same thing



Automatic training examples to abstract things like states

Stat id and city ids help with this

Were able to apply framework broadly

Lot of diff logical form “lambda calc” is just one

Nureal nets do poorly o this

Double the error rate

However with augmentation work can get up to a similar performance

Not as good as hand made, but automation means easier for us to deal with

Does not scale well

One major application: generating sequal query

Can get new data base and new query - generate new querys

But how to be sure that columen names will exsistis – pointer methods

**Attention: problems with seq 2 seq modeling**

Attention is an invoation in making stuff actually work – exspecailly for pretrained transformer models

Motivation come from problems with seq2seq

Tend to like to repeate self – might be under trained onereason

You can getstuck in loop

Coverage – translate esch item once

;stms are not super flexible they are trained for a certaub sentence length

So the fina; issue is unknown word

Even if have an embedding, might not be worth

What to solve – word by word translation

Not a good assumption

Not always true

How can achieve without too much effort: attention

Predict a distribution of position of out

Assign weight to a particular location as wellas words

Achive same thing will ;earn how to do this

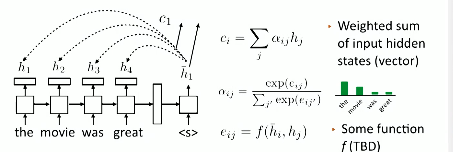
**Attention: model and implementation**

The idea is to predict. Pver source sentence positions

Will help make better prediction

Very use even beyond translation

For each decoddr compute withed sum of input states



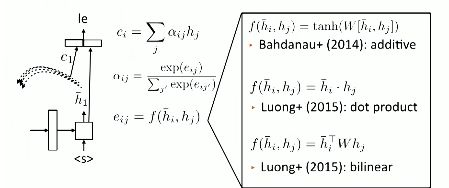
Soft max so that we get a distribution

Get c vector and append with h bar to predict the output

Want to take weighted sum and mult with soft max to get what should be where

So what should we use as our function x?

A non linear layer of a neural net, dot product, or bilinear



This uses out puts of later after they have gone through an lstm

Use cell state and the hidden state - you can put attion layers before or after RNN both work

What can attention do? If attending to position in input that crresonds to output it is esy

* It will copy and subsample the sequence

Lstm can learn to count and create position based addressing

Lstm can learn to reject certain inputs

It is very effect at walking an attention pointer thorug a sentence

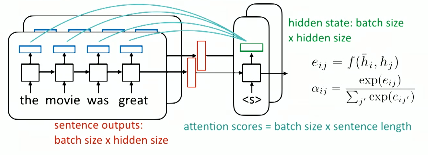
Ca[tures something about identiy o fthe word in context

Decoder is now responsible for selecting what to attend to

No long have to model quite as much in the hidden state

Batching with attention:

Batchsize x sentence length x hidden size -> this is what we get from lstm



Attention scors are batch x sentence so need to broad cast the hidden state

Make sure all of your tensor are actually the correct size

**Copying and Pointer**

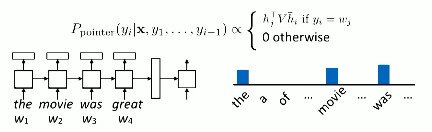
Similar idea of attention but futhers the idea of checking with the input when producing the output

Some words are impossible to predict – they are in the vocab

We can solve this with pointer nextworks

P(vocab) – some prob dirtabution over full vocab

pointer predict based on input words instead of the vocab



For distribution over sources iencdoed wording

Take distribution over what can generate

Can combine attion and oitner

Call a mixer models with indermeate pointers

Model can dynamically switch from attion to pointer

Take input vocab and for each example add tokens that don’t exsist to new tokens

This allows us to place distribution on these words as well

The copy mechanism provode forunknown words

Difference of 7-10% which is a huge improvement

Less emphasis on it these days – now we break up words to understand them using attentions

**Word Piece and Byte Pair encoding**

How to handel rare words by breaking them into smaller pieces

They are sometimes difficult to work with

How big is the vocab – too large gets slow

So can do character level modeling – but these are too small and they don’t have a ton of meaning

So break words into parts ecotax -> eco tax

Add beging of word tokens

Can do a lot with this idea

Byte pari: create common pairs – the merge with other common characher pairs

i-n might be a pair or t-h

a lot of merges - vocab in the thousands or tens od thoughs d

then might break on re-fuel

how can finad sub word tokens that can be model

google make dataset – how they do google translate

unigram lm is better that byte pair encoding

how t. keep modling paradgimn built up

words can be reppped as sequence of sub words

Transoformers