COMP 790.139 (Fall 2016)
Natural Language Processing
(with some vision, robotics, and deep learning)

Sep 7, 2016

Mohit Bansal

(various slides adapted/borrowed from courses by Dan Klein, Richard Socher, Chris Manning, ACL2016 NMT Tutorial)

Announcements

- Paper reading list on website
- I sent an email asking for top-3 choices
- Please send me choices by tomorrow 09/08 and I will create an assignment
- Most topics will be presented in groups of two discussion leaders

NLP Basics and Core Tasks 2

- Compositional Semantics
- Sentiment Analysis
- Language Modeling
- Machine Translation
- Q&A and other Generation topics, e.g., Dialogue, Summarization will be covered separately in next weeks

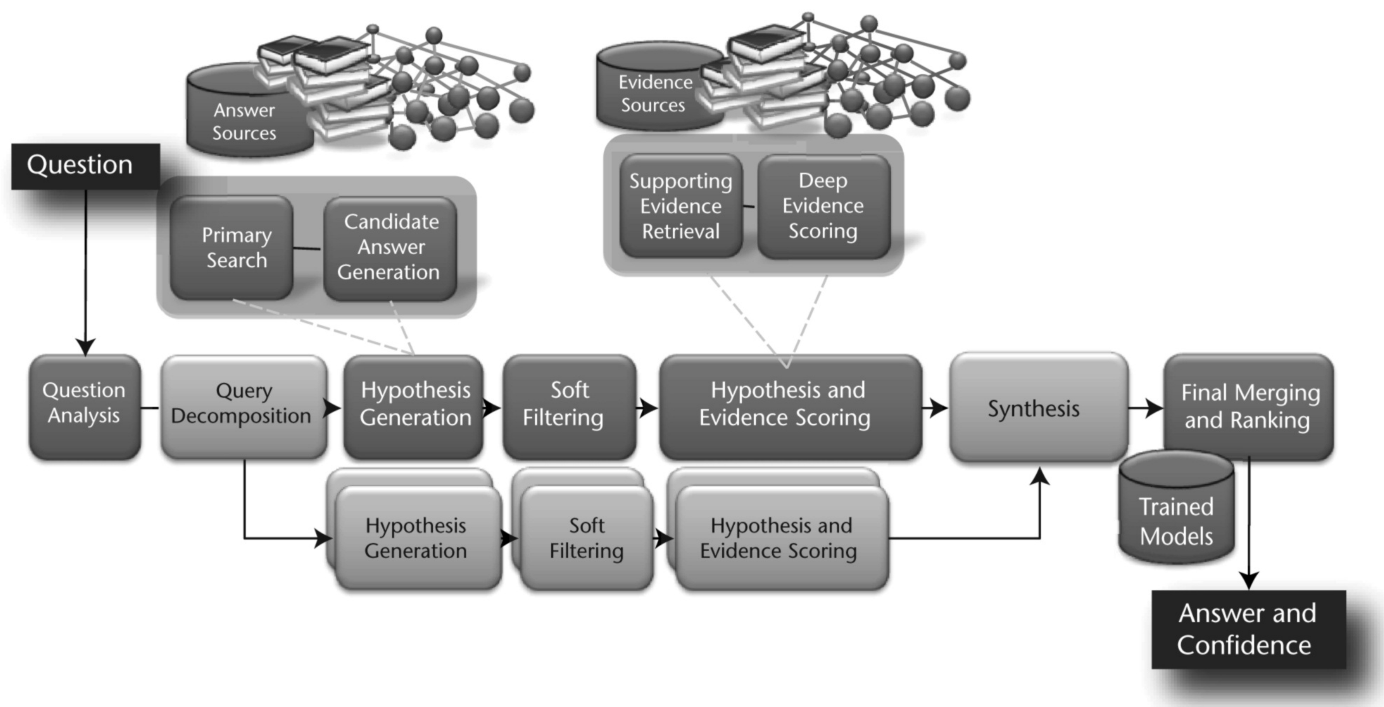
Note: we will be covering some of these briefly (so as to be able to reach the paper reading weeks quickly), so definitely follow up for more details in the prescribed readings and references, and talk to me in office hours!

Compositional Semantics I: Logic form

- Logic-form based, Semantic Parsing
- Useful for Q&A, IE, grounding, comprehension tasks (summarization, reading tasks)
- A lot of focus has been on KB-based Question Answering in this direction

Question Answering

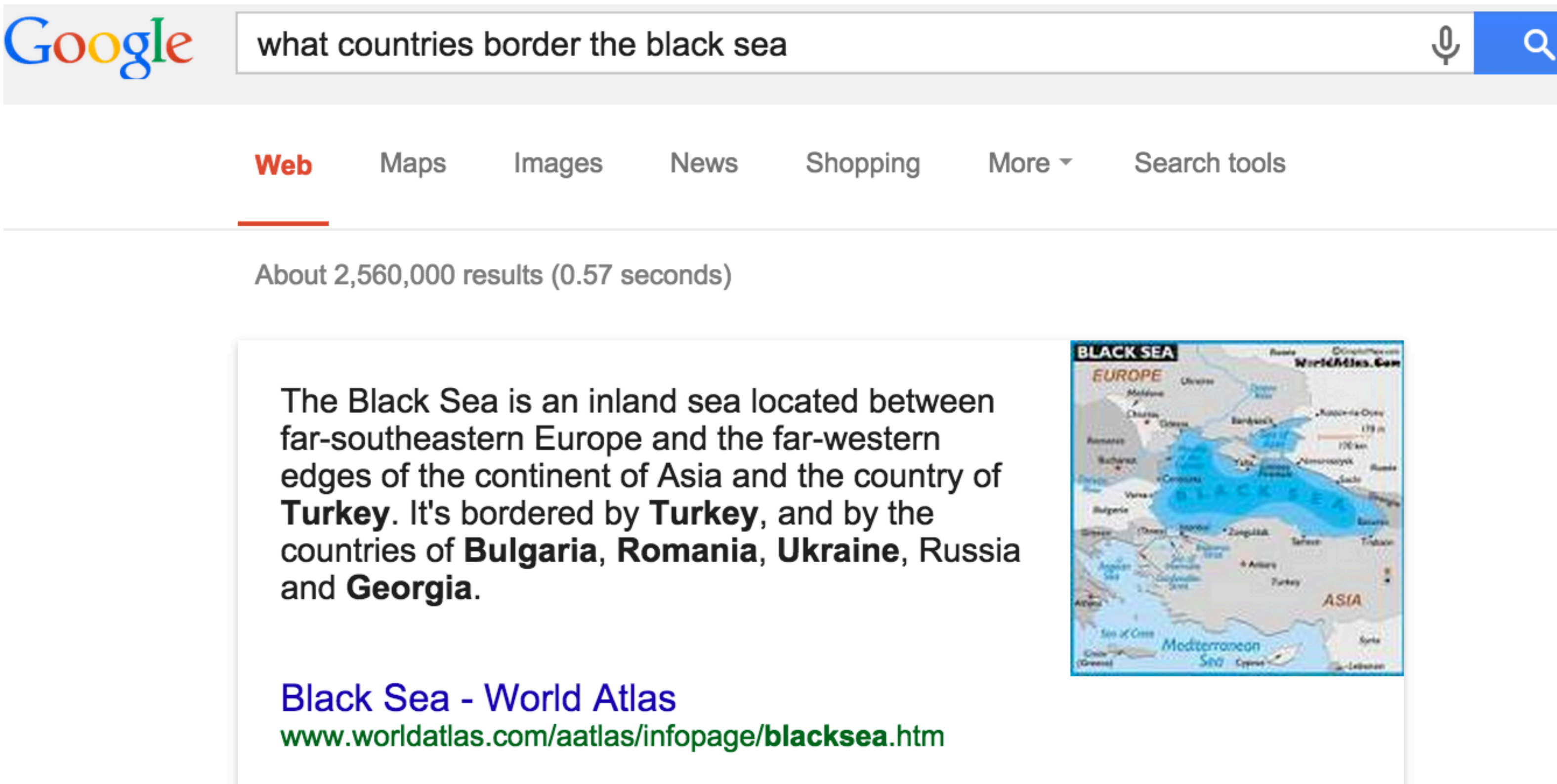
- Initial approaches to Q&A: pattern matching, pattern learning, query rewriting, information extraction
- Next came a large-scale, open-domain IE system like IBM Watson



[Ferrucci et al., 2010]

Deep Q&A: Semantic Parsing

- Complex, free-form, multi-clause questions



Deep Q&A: Semantic Parsing

- Complex, free-form, multi-clause questions

Google | what are the capitals of the countries that border the baltic sea

Web Maps Images News Shopping More Search tools

About 467,000 results (0.46 seconds)

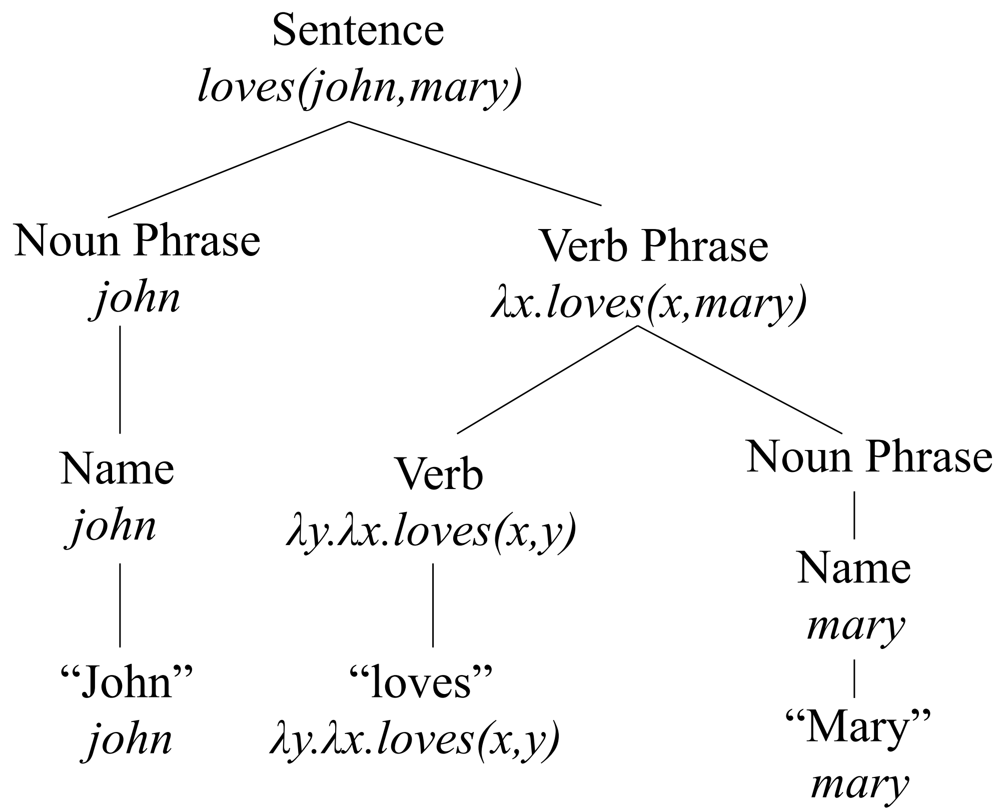
Important cities along the Baltic include:

- The Russian cities of St. Petersburg and Kaliningrad.
- Stockholm, capital of Sweden.
- Copenhagen, capital of Denmark.
- Oslo, capital of Norway.
- Helsinki, capital of Finland.
- Tallinn, capital of Estonia.
- Riga, capital of Latvia.

Baltic Sea - Encyclopedia of Earth
www.eoearth.org/article/**Baltic_Sea**

Semantic Parsing: Logic forms

- Parsing with logic (booleans, individuals, functions) and lambda forms



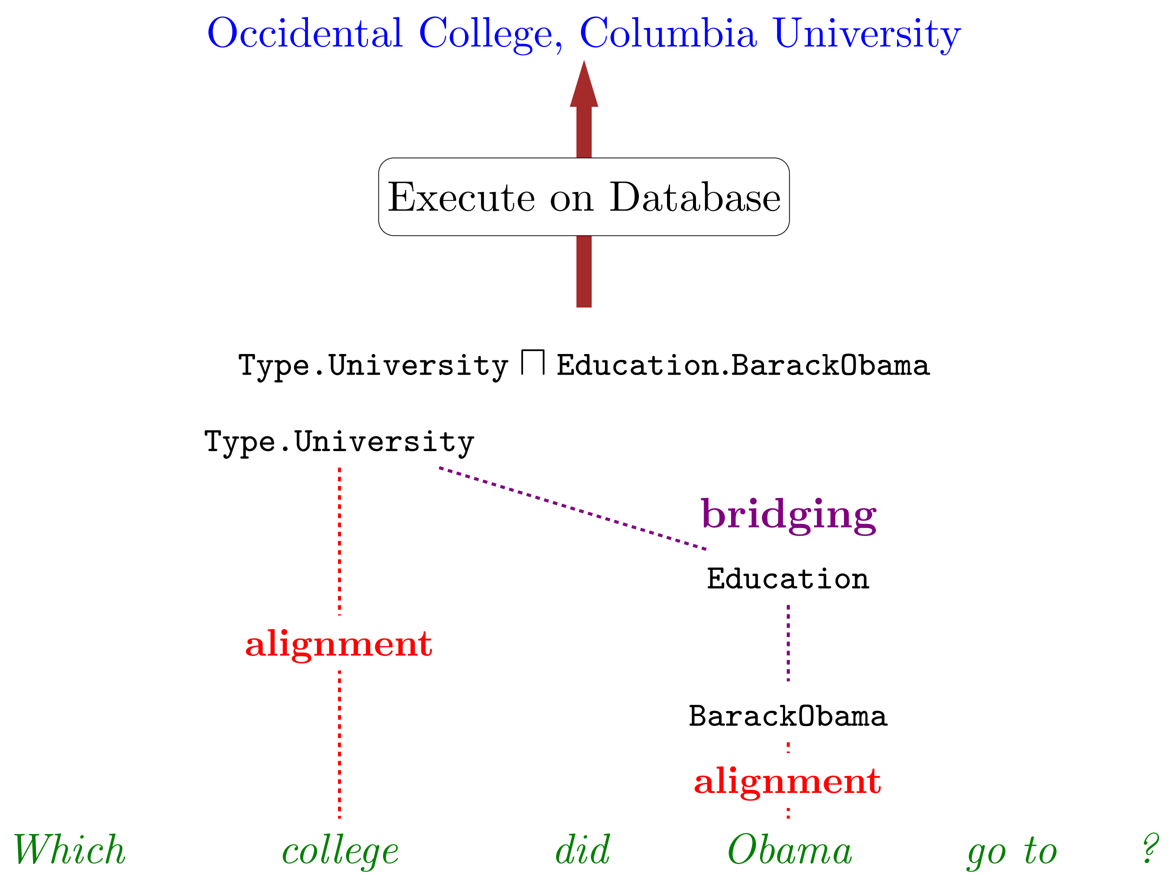
[Wong and Mooney, 2007; Zettlemoyer and Collins, 2007; Poon and Domingos, 2009; Artzi and Zettlemoyer, 2011, 2013; Kwiatkowski et al., 2013; Cai and Yates, 2013; Berant et al., 2013; Poon 2013; Berant and Liang, 2014; Iyyer et al., 2014]

Semantic Parsing Ideas

- Various recent ideas/extensions:
  - unsupervised SP (clustering lambda forms)
  - grounded USP (via databases)
  - Dependency-based compositional semantics (DCS)
  - CCG
  - Bootstrapping w/ conversations
  - On-the-fly ontology matching
  - Question answering on Freebase
  - Paraphrasing

[Wong and Mooney, 2007; Zettlemoyer and Collins, 2007; Poon and Domingos, 2009;
Artzi and Zettlemoyer, 2011, 2013; Kwiatkowski et al., 2013; Cai and Yates, 2013;
Berant et al., 2013; Poon 2013; Berant and Liang, 2014; Iyyer et al., 2014; Yao and Van Durne, 2014]

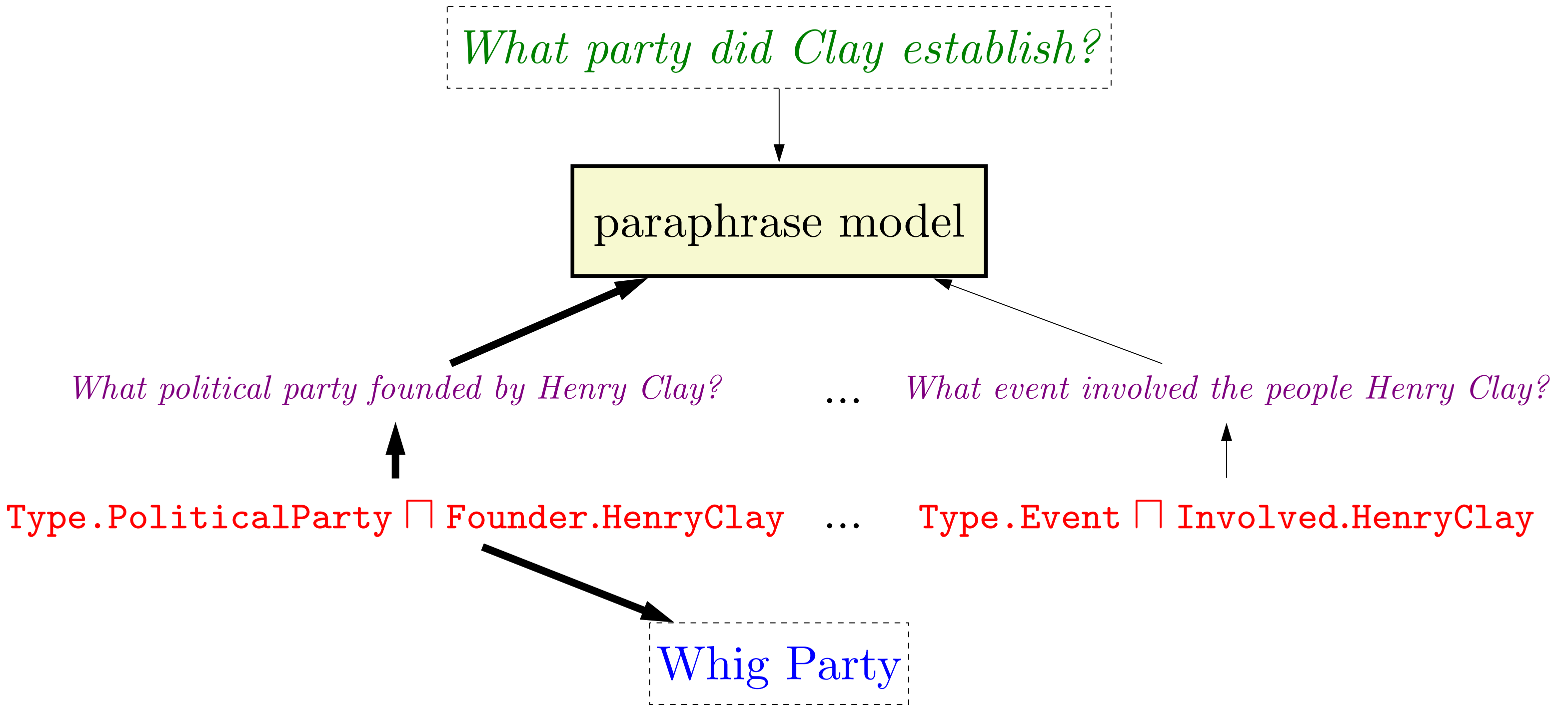
Semantic Parsing on Freebase



Mapping questions to answers via latent logical forms. To narrow down the logical predicate space, they use a (i) coarse *alignment* based on Freebase and a text corpus and (ii) a *bridging* operation that generates predicates compatible with neighboring predicates.

[Berant et al., 2013]

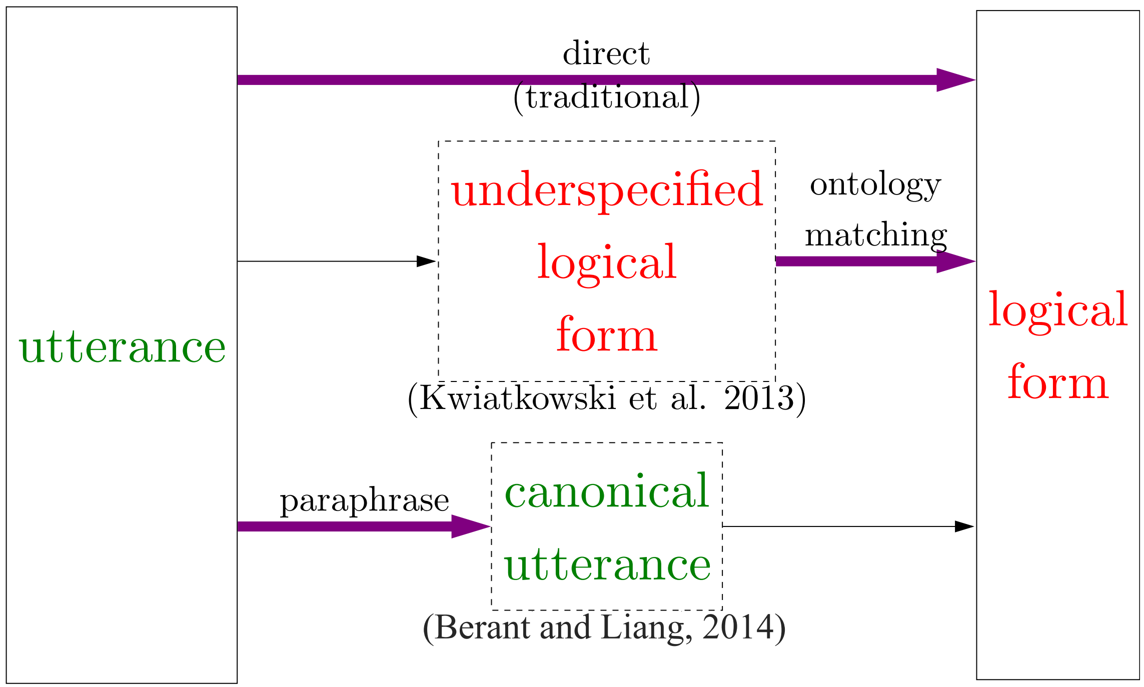
Semantic Parsing via Paraphrasing



For each candidate logical form (red), they generate canonical utterances (purple). The model is trained to paraphrase the input utterance (green) into the canonical utterances associated with the correct denotation (blue).

[Berant and Liang, 2014]

Semantic Parsing via Ontology Matching



The main challenge in semantic parsing is the mismatch between language and the knowledge base. (a) Traditional: map utterances directly to logical forms, (b) Kwiatkowski et al. (2013): map utterance to intermediate, underspecified logical form, then perform ontology matching to handle the mismatch, (c) Berant and Liang (2014): generate intermediate, canonical text utterances for logical forms, then use paraphrase models.

[Kwiatkowski et al., 2013; Berant and Liang, 2014]

Compositional Semantics II: NNs

- Composing, combining word vectors to representations for longer units: phrases, sentences, paragraphs, …

- Initial approaches: point-wise sum, multiplication
  [Mitchell and Lapata, 2010; Blacoe and Lapata, 2012]

- Vector-matrix compositionality [Baroni and Zamparelli, 2010; Zanzotto et al., 2010; Grefenstette and Sadrzadeh, 2011; Socher et al., 2011; Yessenalina and Cardie, 2011]

- Linguistic information added via say parses in RvNNs
  [Socher et al., 2011b, 2012, 2013a, 2013b, 2014; Hermann and Blunsom, 2013]

- Now: Sequential RNNs (with GRU/LSTM gates)

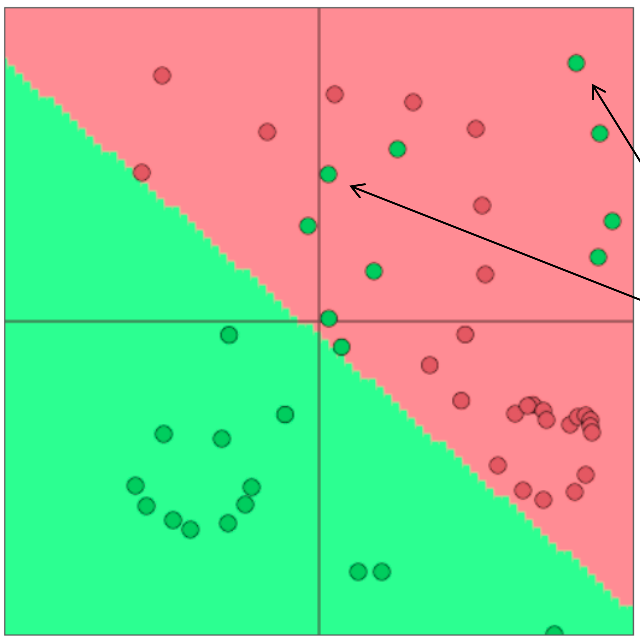
  (Simple vector averaging w/ updating sometimes competitive)

Compositional Semantics II: NNs

- Feed-forward NNs with back-propagation

Softmax (= logistic regression) is not very powerful

- Softmax only linear decision boundaries



→ Lame when problem is complex

Wouldn't it be cool to get these correct?

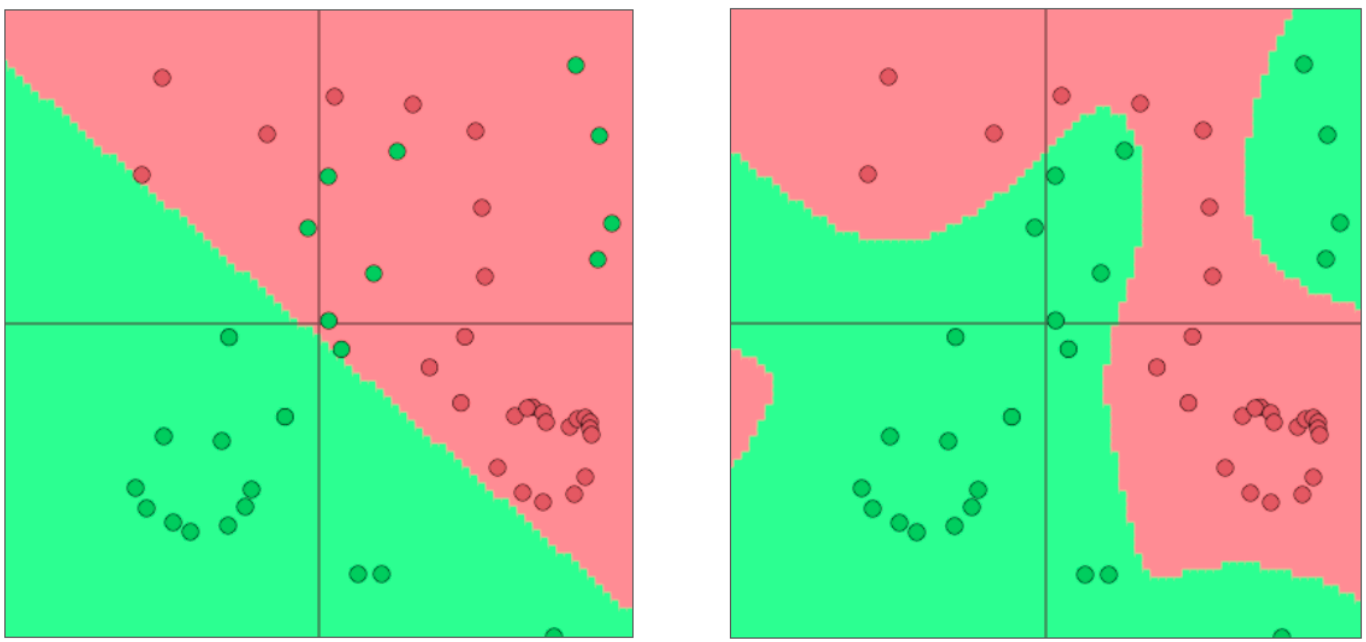
NN and backprop slides from CS224d – Richard Socher

Compositional Semantics II: NNs

- Feed-forward NNs with back-propagation

Neural Nets for the Win!

- Neural networks can learn much more complex functions and nonlinear decision boundaries!



Compositional Semantics II: NNs

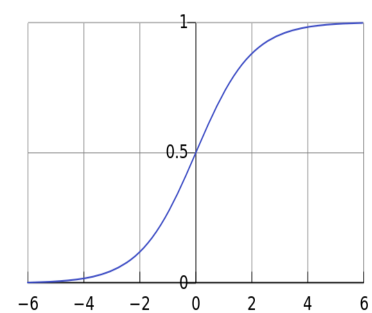
- Feed-forward NNs with back-propagation

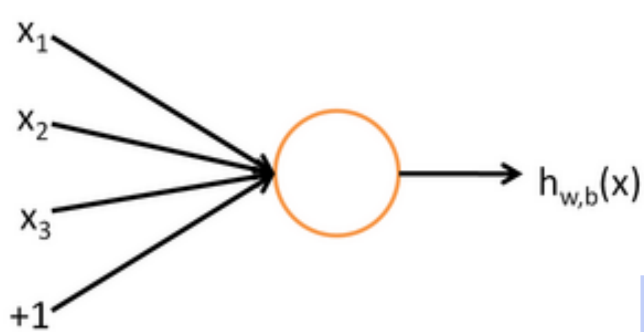
A neuron is essentially a binary logistic regression unit

$$h_{w,b}(x) = f(w^{\top}x + b)$$

b: We can have an "always on" feature, which gives a class prior, or separate it out, as a bias term

$$f(z) = \frac{1}{1+e^{-z}}$$





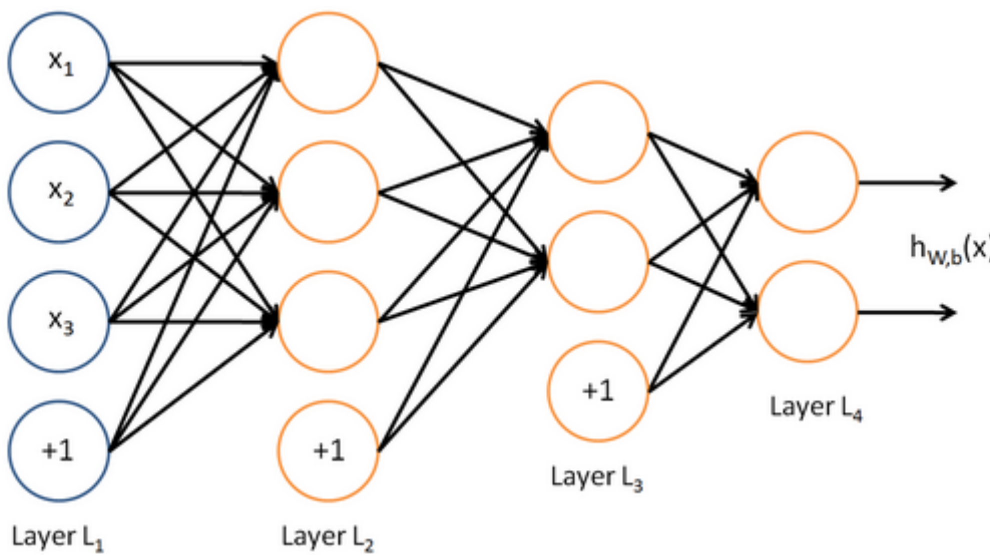
w, *b* are the parameters of this neuron i.e., this logistic regression model

Compositional Semantics II: NNs

- Feed-forward NNs with back-propagation

A neural network
= running several logistic regressions at the same time

Before we know it, we have a multilayer neural network….



Compositional Semantics II: NNs

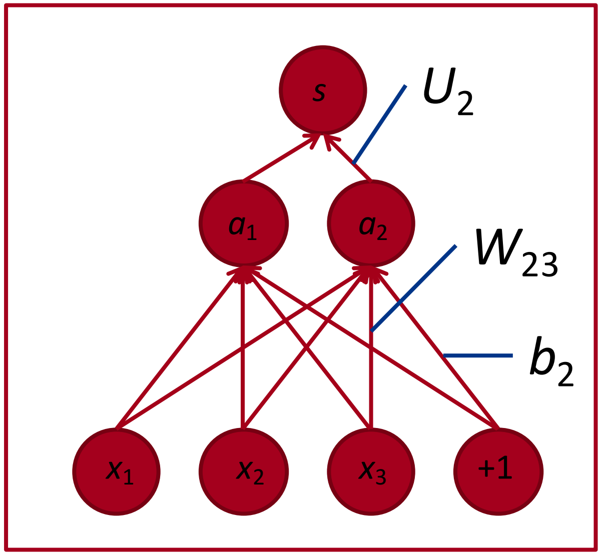
- Feed-forward NNs with back-propagation

Training with Backpropagation

- Let's consider the derivative of a single weight W_{ij}

$$\frac{\partial s}{\partial W} = \frac{\partial}{\partial W} U^T a = \frac{\partial}{\partial W} U^T f(z) = \frac{\partial}{\partial W} U^T f(Wx + b)$$

- This only appears inside a_i
- For example: W_{23} is only used to compute a_2



Compositional Semantics II: NNs

- Feed-forward NNs with back-propagation

Training with Backpropagation

$$\frac{\partial s}{\partial W} = \frac{\partial}{\partial W} U^T a = \frac{\partial}{\partial W} U^T f(z) = \frac{\partial}{\partial W} U^T f(Wx + b)$$

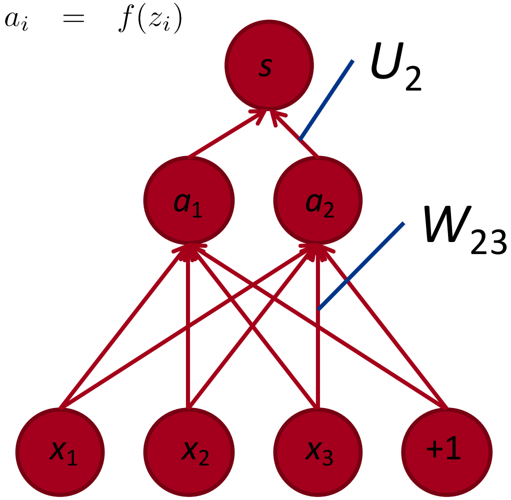
Derivative of weight W_{ij}:

$$\frac{\partial y}{\partial x} = \frac{\partial y}{\partial u} \frac{\partial u}{\partial x}$$

$$\frac{\partial}{\partial W_{ij}} U^T a \quad \rightarrow \quad \frac{\partial}{\partial W_{ij}} U_i a_i$$

$$\begin{aligned} U_i \frac{\partial}{\partial W_{ij}} a_i &= U_i \frac{\partial a_i}{\partial z_i} \frac{\partial z_i}{\partial W_{ij}} \\ &= U_i \frac{\partial f(z_i)}{\partial z_i} \frac{\partial z_i}{\partial W_{ij}} \\ &= U_i f'(z_i) \frac{\partial z_i}{\partial W_{ij}} \\ &= U_i f'(z_i) \frac{\partial W_{i.}x + b_i}{\partial W_{ij}} \end{aligned}$$

$$\begin{aligned} z_i &= W_{i.}x + b_i &= \sum_{j=1}^{3} W_{ij}x_j + b_i \\ a_i &= f(z_i) \end{aligned}$$



Compositional Semantics II: NNs

- Feed-forward NNs with back-propagation

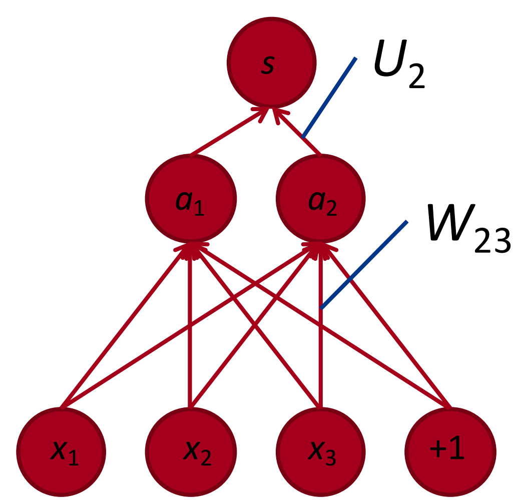
Training with Backpropagation

Derivative of single weight W_{ij} :

$$z_i = W_{i\cdot}x + b_i = \sum_{j=1}^{3} W_{ij}x_j + b_i$$
$$a_i = f(z_i)$$

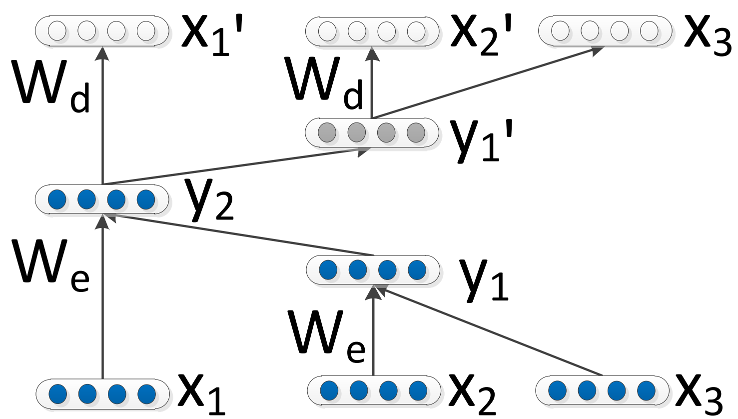
$$\begin{aligned}
U_i \frac{\partial}{\partial W_{ij}} a_i &= U_i f'(z_i) \frac{\partial W_{i\cdot}x + b_i}{\partial W_{ij}} \\
&= U_i f'(z_i) \frac{\partial}{\partial W_{ij}} \sum_k W_{ik}x_k \\
&= \underbrace{U_i f'(z_i)} x_j \\
&= \underbrace{\delta_i}_{\text{Local error signal}} \quad \underbrace{x_j}_{\text{Local input signal}}
\end{aligned}$$

where $f'(z) = f(z)(1 - f(z))$ for logistic f



Syntactically Recursive Autoencoders

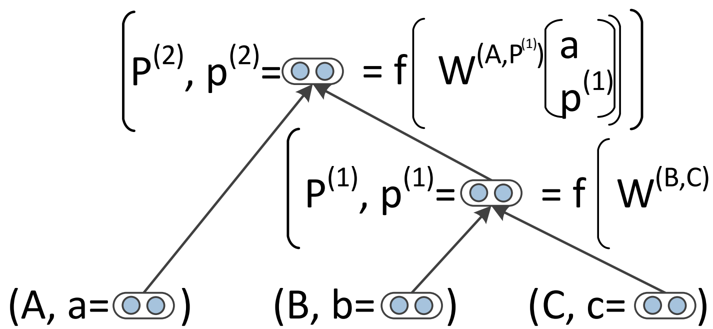
- Socher et al., 2011: Recursive autoencoders (unsupervised) on constituent parse trees

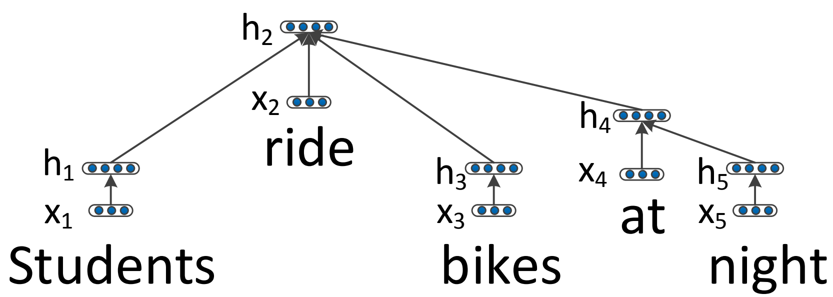


- The unfolding autoencoder which tries to reconstruct all leaf nodes underneath each node.

Syntactically Recursive NNs

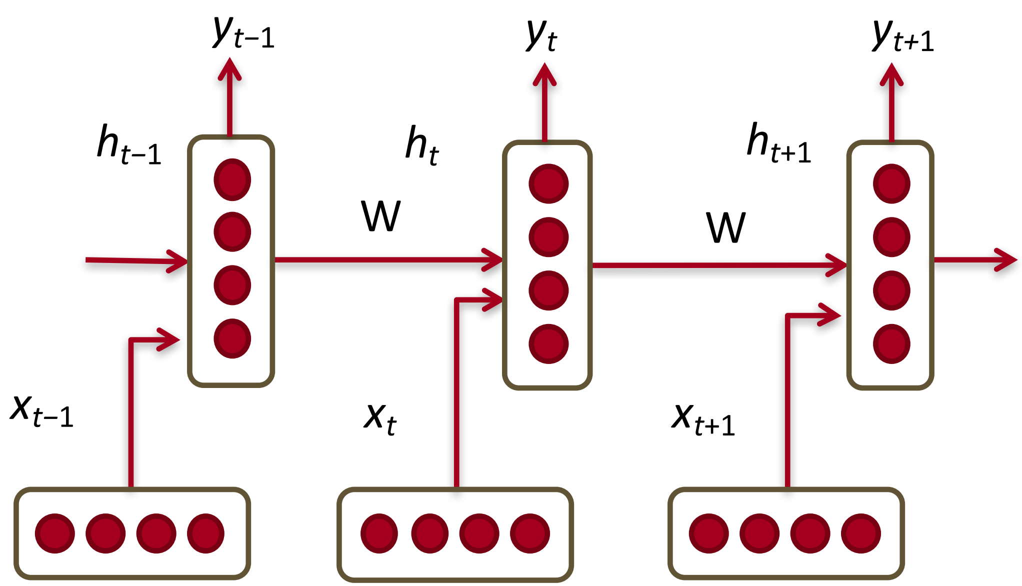
- Socher et al., 2013a, 2014: RvNNs on constituent and dependency parse trees





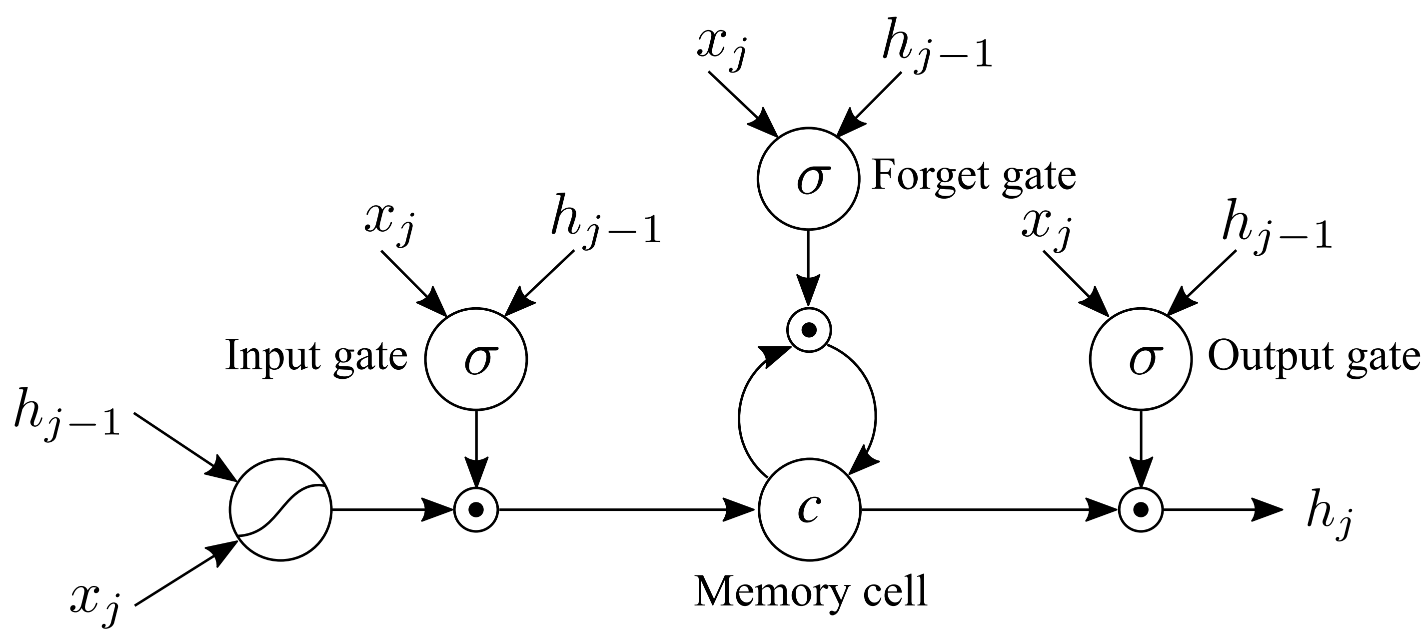
Recurrent NNs

- Recurrent NNs (RNNs) are non-tree, sequential versions of RvNNs
- Weights tied together for each time step



LSTM RNNs

- LSTM (Long short term memory) RNNs have gates for forgetting, allowing learning of longer-term connections by avoiding vanishing/ exploding gradients



Various Applications of such RNNs

- Classification: Sentiment Analysis
- Language Modeling
- Generation: End-to-end MT
- Others: Parsing, Captioning, Summarization, Q&A, Dialogue (some will be covered in future weeks)
- Demos: http://deeplearning.net/demos/, http://cs.stanford.edu/people/karpathy/deepimagesent/rankingdemo/, https://www.metamind.io/

Sentiment Analysis

Sentiment Analysis with Python NLTK Text Classification

This is a demonstration of **sentiment analysis** using a `NLTK 2.0.4` powered **text classification** process. It can tell you whether it thinks the text you enter below expresses **positive sentiment**, **negative sentiment**, or if it's **neutral**. Using **hierarchical classification**, *neutrality* is determined first, and *sentiment polarity* is determined second, but only if the text is not neutral.

Analyze Sentiment

Language

english

Enter text

```
It always amazes me how Universal never cares to create anything remotely clever when it comes to their animations, and so once again they come up with a harmless little story that wants to be cute and funny (which it is sometimes) but is only bound to be quickly forgotten.
```

Enter up to 50000 characters

Analyze

Sentiment Analysis Results

The text is **neg**.

The final sentiment is determined by looking at the classification probabilities below.

Subjectivity

- neutral: 0.3
- **polar: 0.7**

Polarity

- pos: 0.2
- **neg: 0.8**

Sentiment Analysis

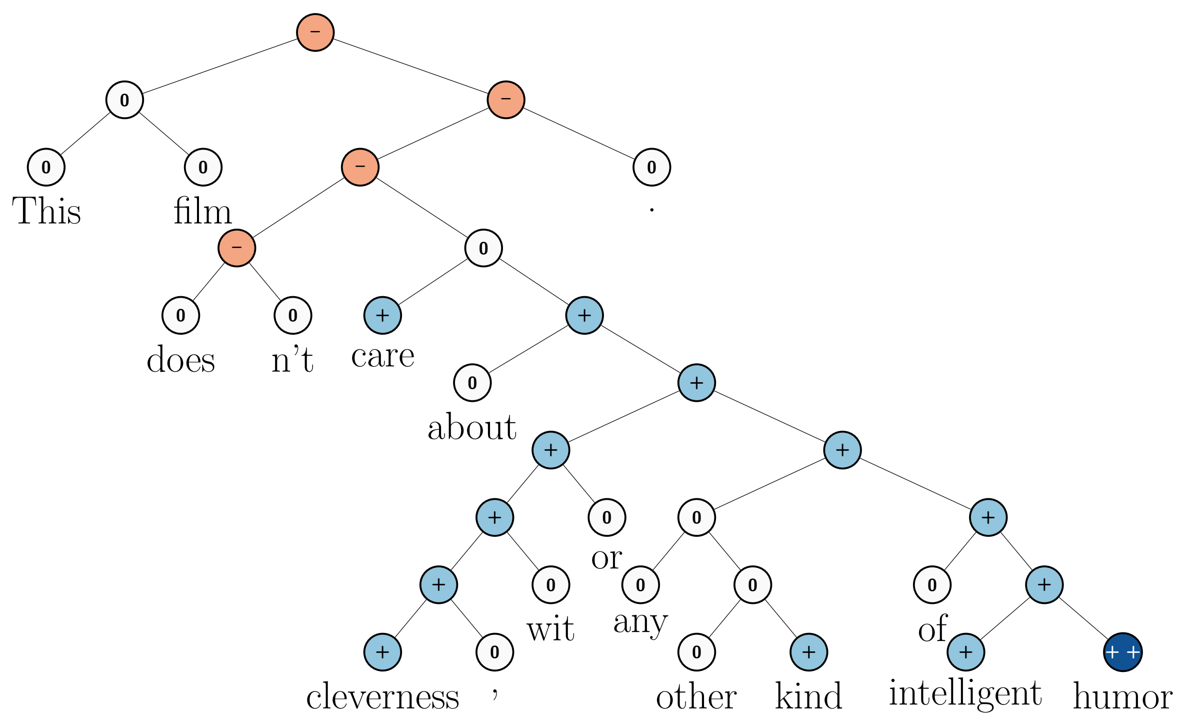
- Earlier methods used bag of words, e.g., lexicons of positive and negative words and phrases
- Cannot distinguish tricky cases like:

\+ *white blood cells destroying an infection*
– *an infection destroying white blood cells*

\+ *There are slow and repetitive parts but it has just enough spice to keep it interesting.*
– *Stealing Harvard doesn't care about cleverness, wit or any other kind of intelligent humor.*

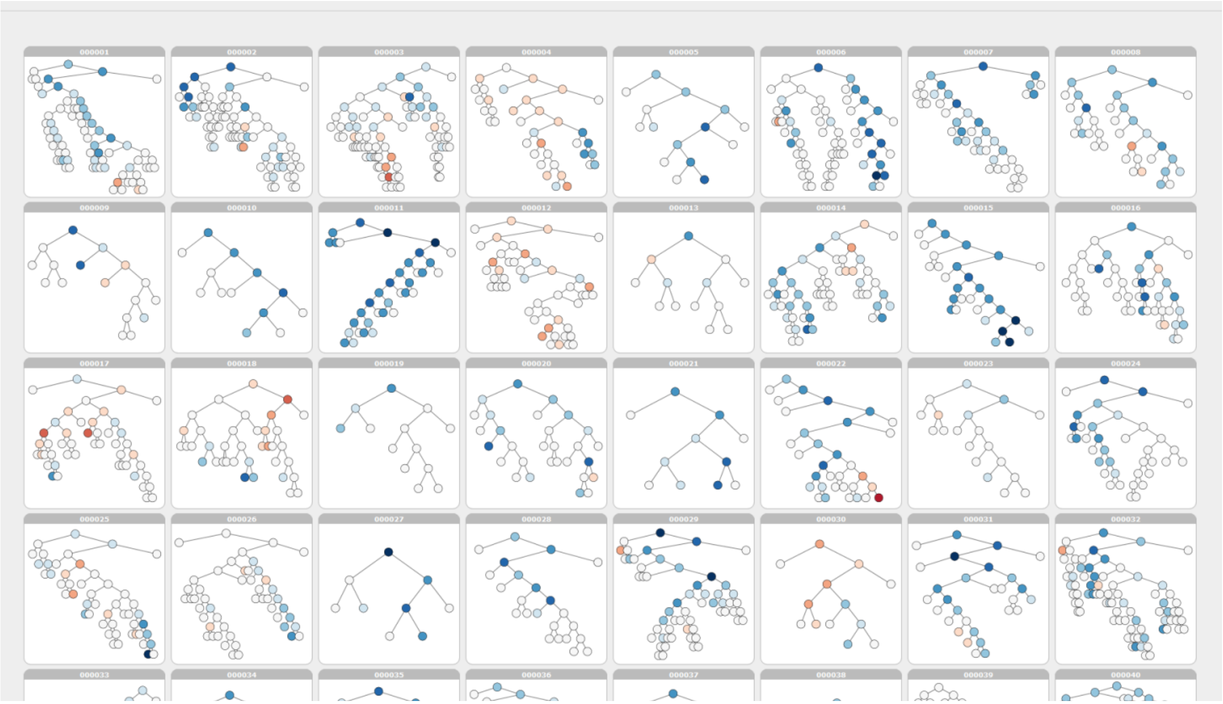
Sentiment Analysis

- Even simpler issues like negation hard to understand
- Socher et al., 2013b present new compositional training data and new composition model



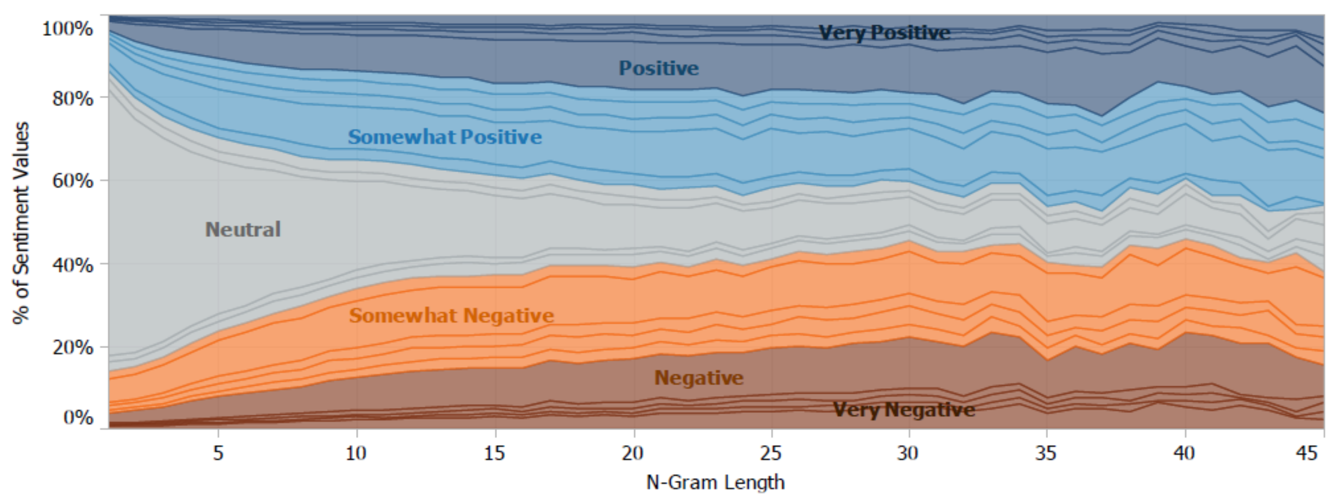
Sentiment Analysis

- Even simpler issues like negation hard to understand
- Socher et al., 2013b present new compositional training data and new composition model



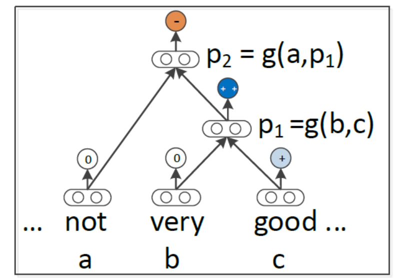
Sentiment Analysis

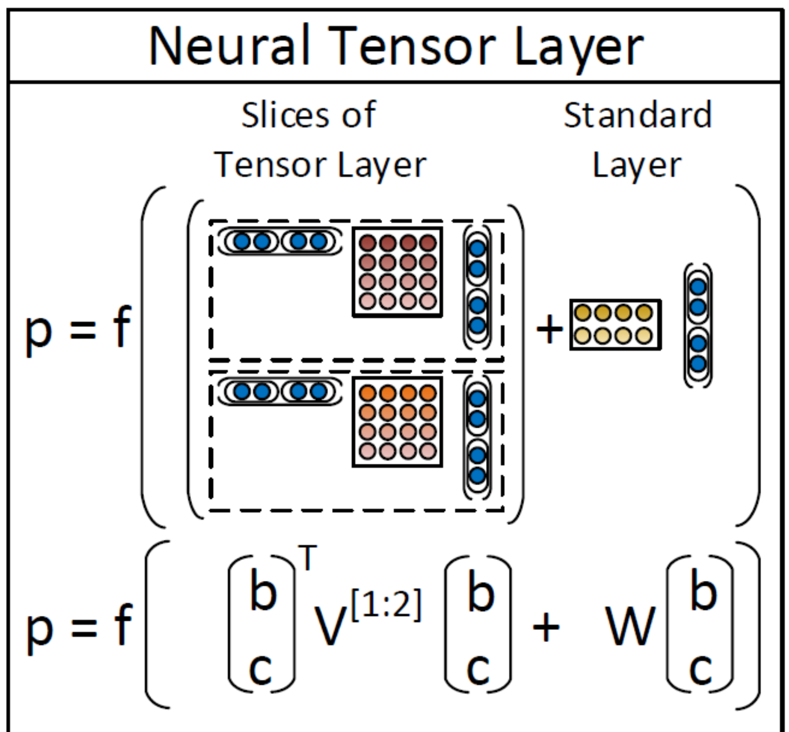
- Sentiment Compositionality:
  - Parse trees of 11,855 sentences
  - 215,154 phrases with labels
  - Allows training and evaluating with compositional information



Sentiment Analysis

- Better Models: Recursive Neural Tensor Network (RNTN)



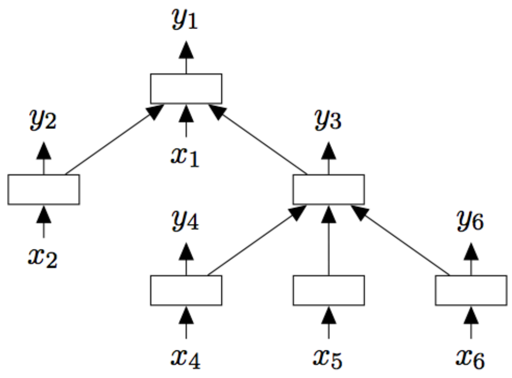


Sentiment Analysis

- Better Models: Tree-based LSTM-RNNs

Tree LSTMs

- We can use those ideas in grammatical tree structures!
- Paper: Tai et al. 2015:
  Improved Semantic Representations From Tree-Structured Long Short-Term Memory Networks



- Idea: Sum the child vectors in a tree structure
- Each child has its own forget gate
- Same softmax on h

$$\boxed{\tilde{h}_j = \sum_{k \in C(j)} h_k,}$$

$$i_j = \sigma\left(W^{(i)} x_j + U^{(i)} \boxed{\tilde{h}_j} + b^{(i)}\right),$$

$$\boxed{f_{jk}} = \sigma\left(W^{(f)} x_j + U^{(f)} h_k + b^{(f)}\right),$$

$$o_j = \sigma\left(W^{(o)} x_j + U^{(o)} \boxed{\tilde{h}_j} + b^{(o)}\right),$$

$$u_j = \tanh\left(W^{(u)} x_j + U^{(u)} \tilde{h}_j + b^{(u)}\right),$$

$$c_j = i_j \odot u_j + \boxed{\sum_{k \in C(j)} f_{jk} \odot c_k,}$$

$$h_j = o_j \odot \tanh(c_j),$$

Sentiment Compositionality

Results on Stanford Sentiment Treebank

Method	Fine-grained	Binary
RAE (Socher et al., 2013)	43.2	82.4
MV-RNN (Socher et al., 2013)	44.4	82.9
RNTN (Socher et al., 2013)	45.7	85.4
DCNN (Blunsom et al., 2014)	48.5	86.8
Paragraph-Vec (Le and Mikolov, 2014)	48.7	87.8
CNN-non-static (Kim, 2014)	48.0	87.2
CNN-multichannel (Kim, 2014)	47.4	**88.1**
DRNN (Irsoy and Cardie, 2014)	49.8	86.6
LSTM	45.8	86.7
Bidirectional LSTM	49.1	86.8
2-layer LSTM	47.5	85.5
2-layer Bidirectional LSTM	46.2	84.8
Constituency Tree LSTM (no tuning)	46.7	86.6
Constituency Tree LSTM	**50.6**	86.9

of word vectors

- Demos: http://nlp.stanford.edu:8080/sentiment/rntnDemo.html

[Yessenalina and Cardie, 2011; Socher et al., 2013b]

Language Modeling

- A language model is a distribution over sequences of words (sentences)

$$P(\mathbf{w}) = P(w_1 \ldots w_n)$$

- Purpose it to usually assign high weights to plausible sentences, e.g., in speech recognition or machine translation

- Also used for language generation now (predict next word given previous words), esp. w/ new RNN models

Traditional N-gram LMs

- Use chain rule to generate words left-to-right

$$P(w_1 \dots w_n) = \prod_i P(w_i | w_1 \dots w_{i-1})$$

- Can't condition on the entire left context

P(??? | Turn to page 134 and look at the picture of the)

- N-gram models make a Markov assumption

$$P(w_1 \dots w_n) = \prod_i P(w_i | w_{i-k} \dots w_{i-1})$$

$$P(\text{please close the door}) =$$

$$P(\text{please}|\text{START})P(\text{close}|\text{please}) \dots P(\text{STOP}|door)$$

Traditional N-gram LMs

- How do we know P(w | history)?
  - Use statistics from data (examples using Google N-Grams)
  - E.g. what is P(door | the)?

Training Counts

```
198015222 the first
194623024 the same
168504105 the following
158562063 the world

14112454 the door
-----------------
23135851162 the *
```

$$\hat{P}(\text{door}|\text{the}) = \frac{14112454}{23135851162}$$

$$= 0.0006$$

- This is the *maximum likelihood* estimate

Sparsity Issue & Parameter Estimation

- New words all the time (*antidisestablishmentarianism*, *kakorrhaphiophobia*,, *www.xyzabc156.com*)….worse for new bigrams and trigrams!
  - Maximum likelihood estimates won't get us very far

$$\hat{P}(w|w_{-1}) = \frac{c(w_{-1}, w)}{\sum_{w'} c(w_{-1}, w')}$$

3516 wipe off the excess
1034 wipe off the dust
547 wipe off the sweat
518 wipe off the mouthpiece
…
120 wipe off the grease
0 wipe off the sauce
0 wipe off the mice

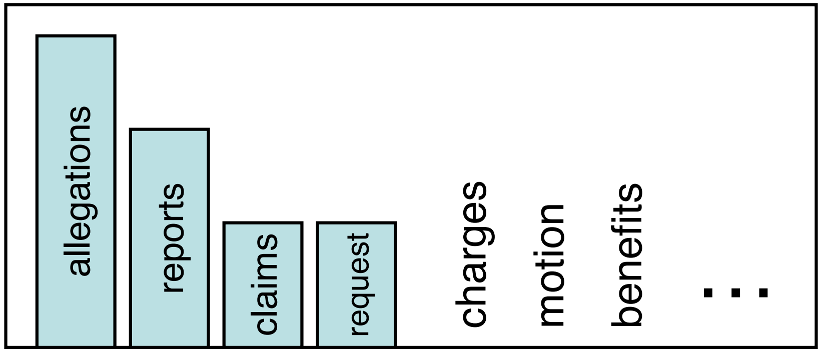
28048 wipe off the *

  - Need to *smooth* these estimates
  - General method (procedurally)
    - Take your empirical counts
    - Modify them in various ways to improve estimates
  - General method (mathematically)
    - Often can give estimators a formal statistical interpretation
    - … but not always
    - Approaches that are mathematically obvious aren't always what works

Smoothing Techniques

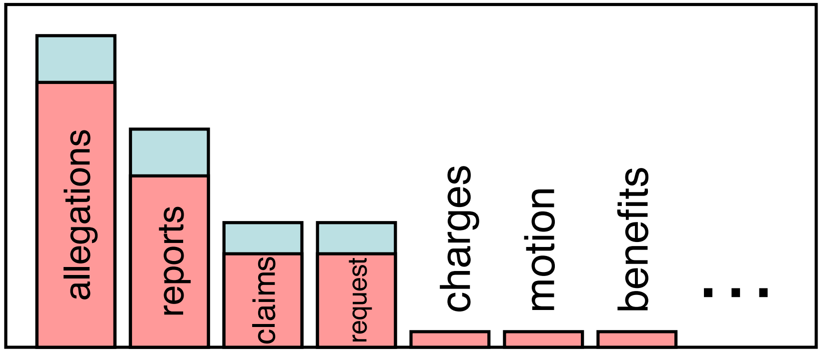
- We often want to make estimates from sparse statistics:

P(w | denied the)
3 allegations
2 reports
1 claims
1 request
7 total



- Smoothing flattens spiky distributions so they generalize better

P(w | denied the)
2.5 allegations
1.5 reports
0.5 claims
0.5 request
2 other
7 total



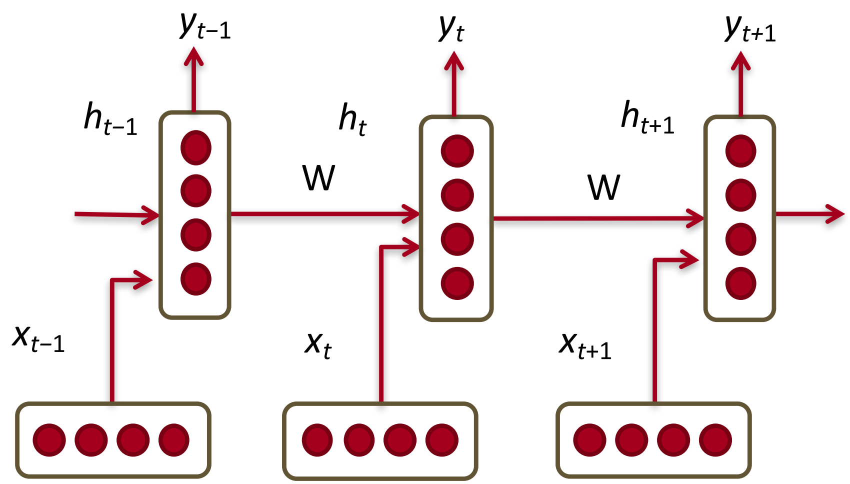
- Very important all over NLP, but easy to do badly!

Smoothing Techniques

- Classic Solution: add-one or add small priors to numer/denom
- Backing off to smaller n-grams
- Held-out Reweighting: Important to optimize/estimate how models generalize! So use held-out data to estimate the map of old count to new count
- Kneser-Ney Discounting: two successful ideas:
  - Idea 1: observed n-grams occur more in training than they will later
  - Idea 2: Type-based fertility (based on how common the word type is)
- Read Chen and Goodman, 1996 for various details and graphs!

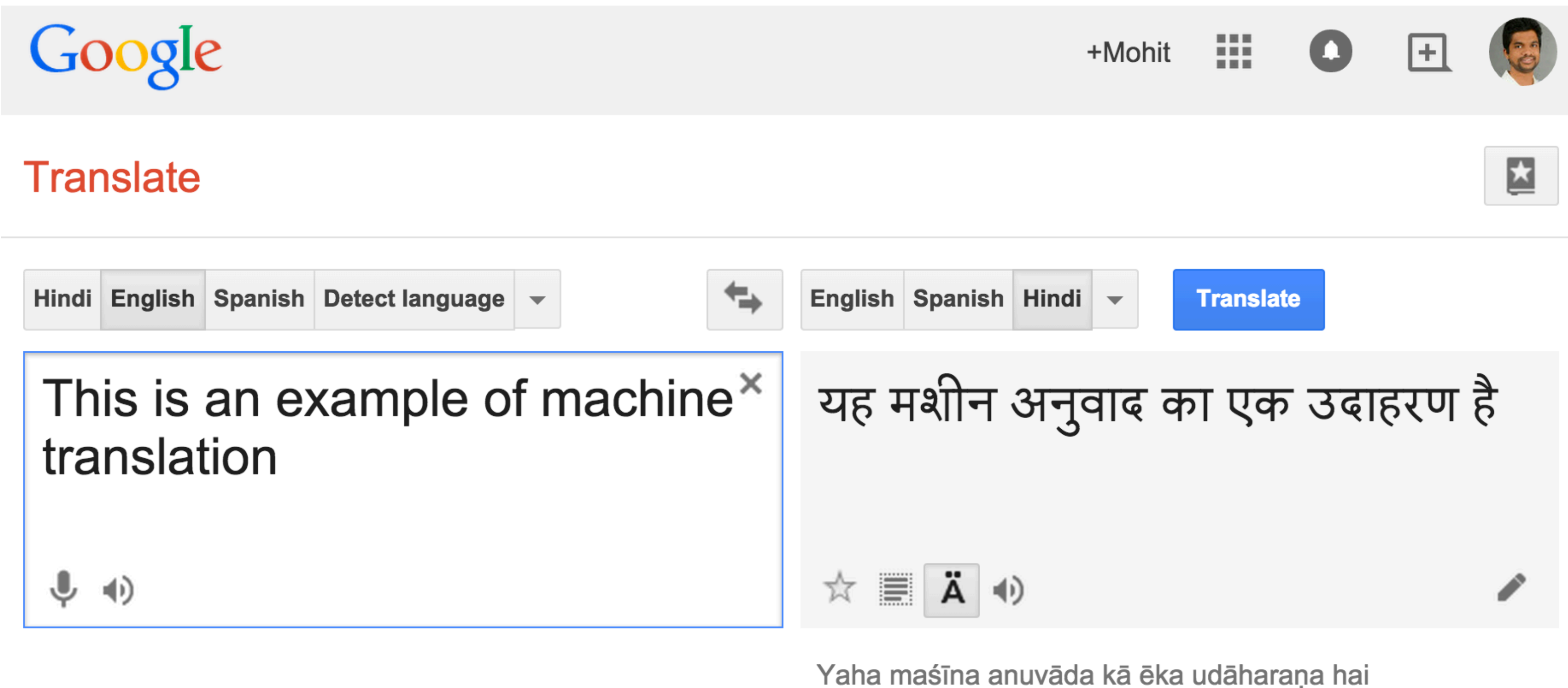
RNN Language Models

- Avoid huge number of n-grams; Memory requirement only scales with #words
- Can condition on all previous history (with forget gates)
- Loss function on identity of predicted word at each time step
- But harder/slower to train and reach optimum?



Machine Translation

- Useful for tons of companies, online traffic, and our international communication!

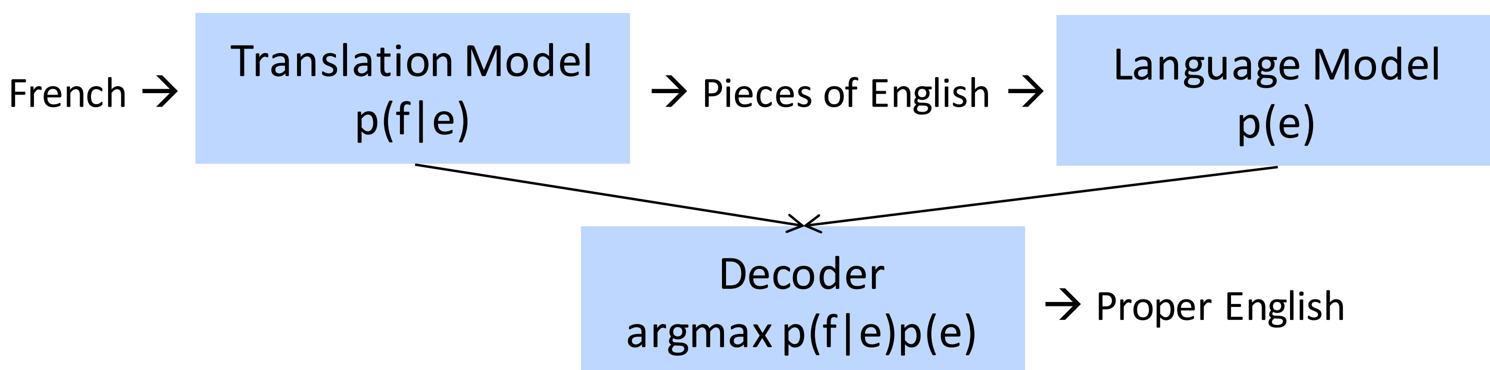


Traditional Machine Translation

- Source language f, e.g. French
- Target language e, e.g. English
- Probabilistic formulation (using Bayes rule)

$$\hat{e} = \text{argmax}_e p(e|f) = \text{argmax}_e p(f|e)p(e)$$

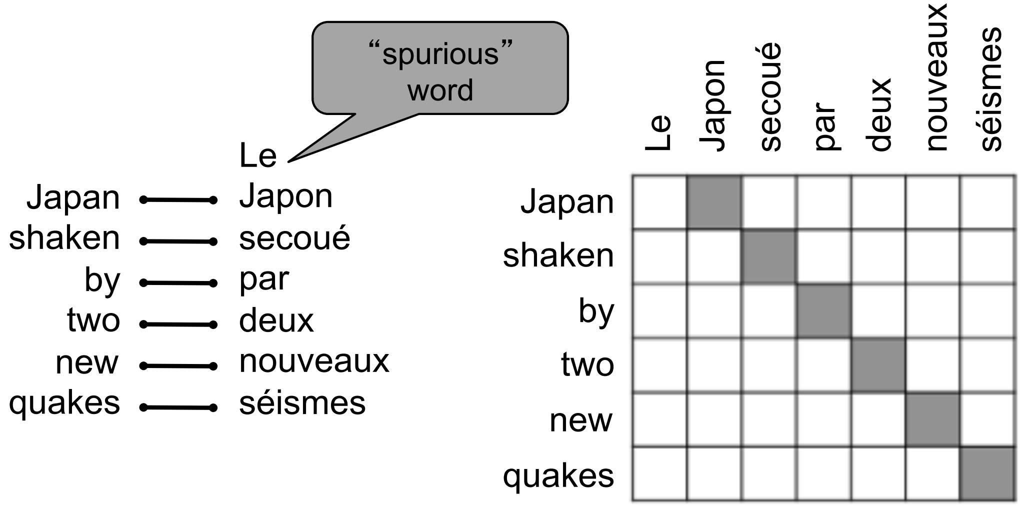
- Translation model p(f|e) trained on parallel corpus
- Language model p(e) trained on English only corpus (lots, free!)



Traditional Machine Translation

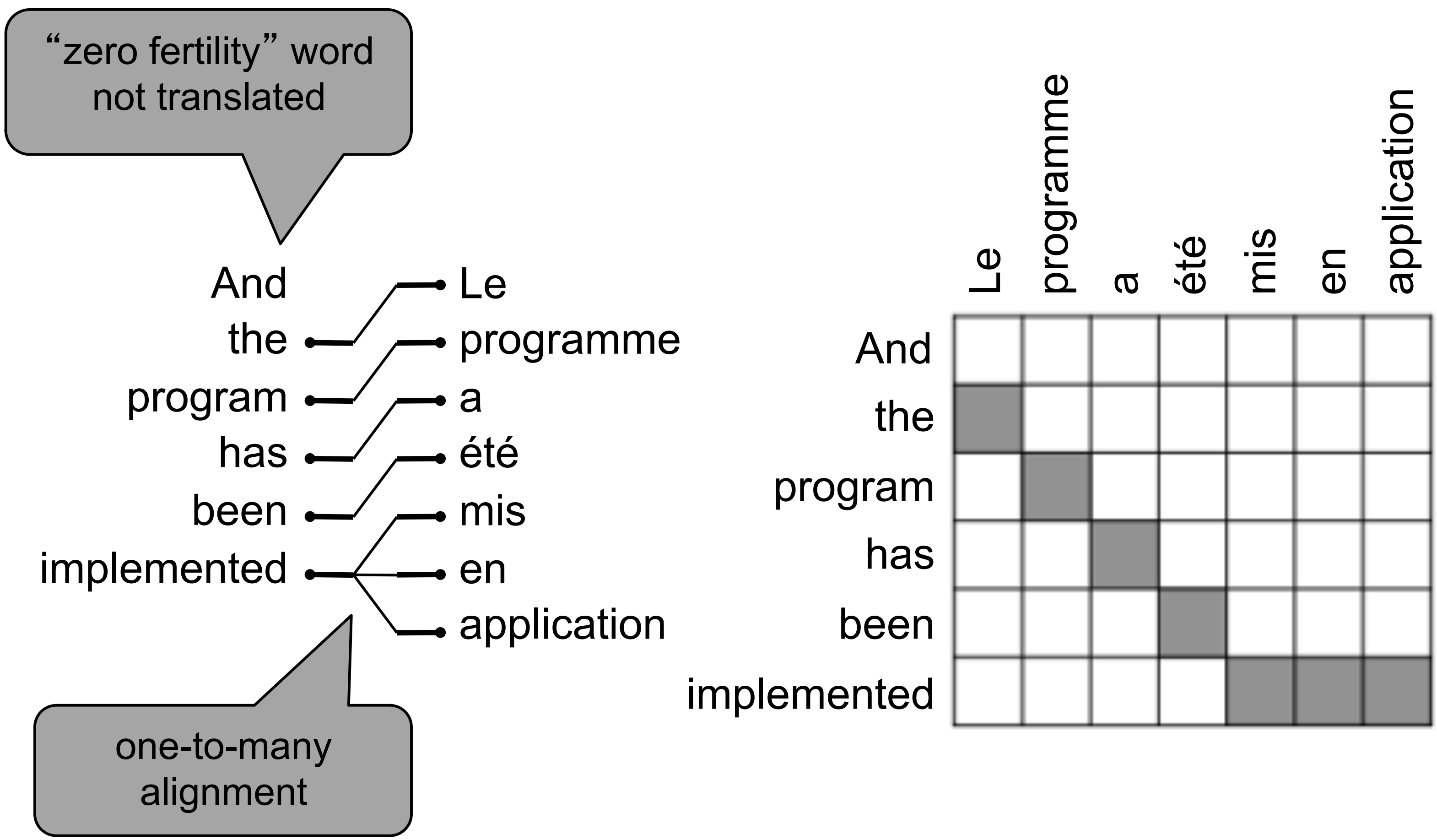
Step 1: Alignment

Goal: know which word or phrases in source language would translate to what words or phrases in target language? → Hard already!



Traditional Machine Translation

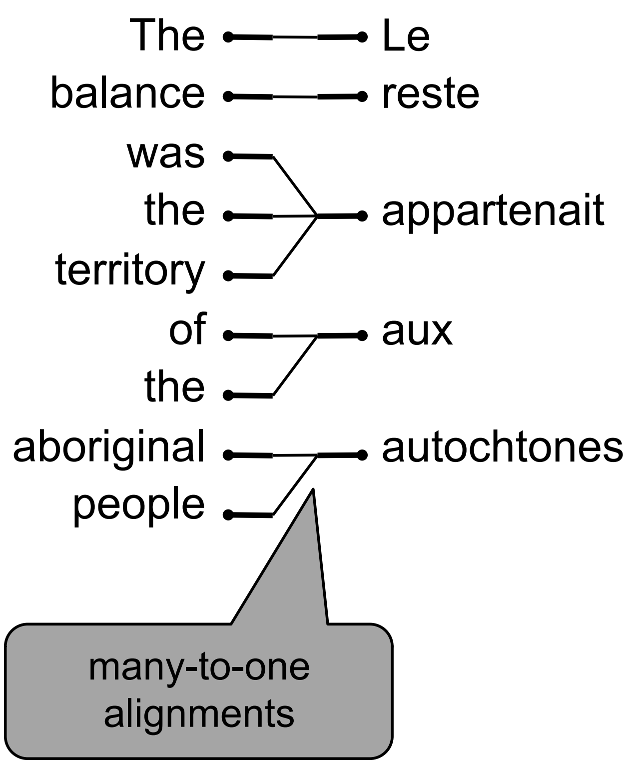
Step 1: Alignment

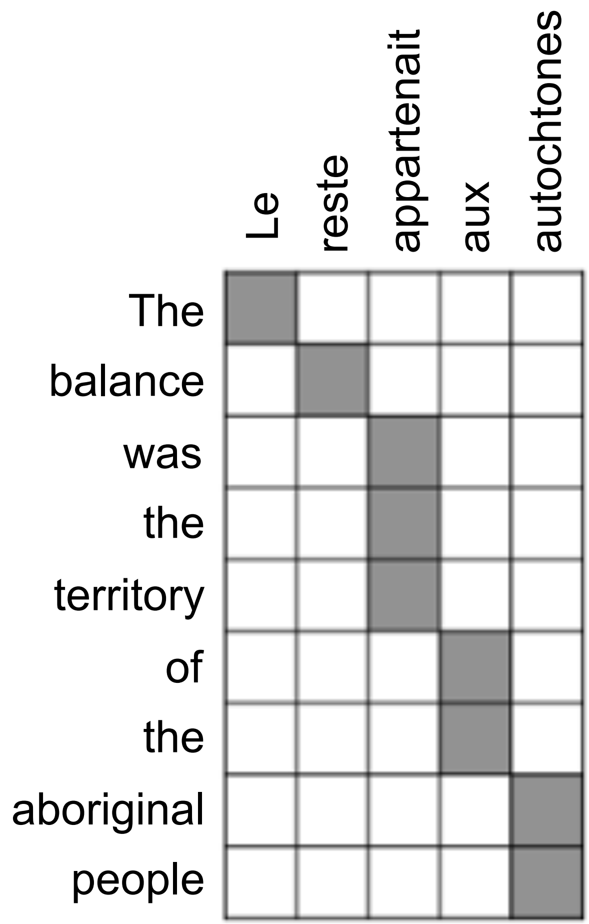


Traditional Machine Translation

Step 1: Alignment

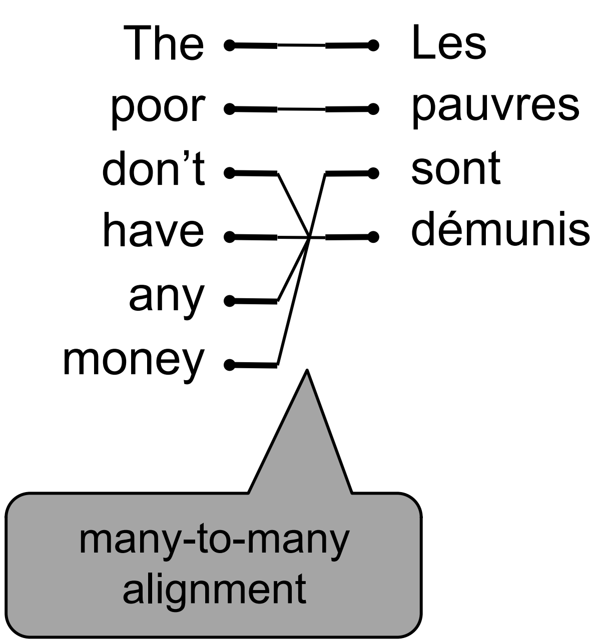
Really hard :/

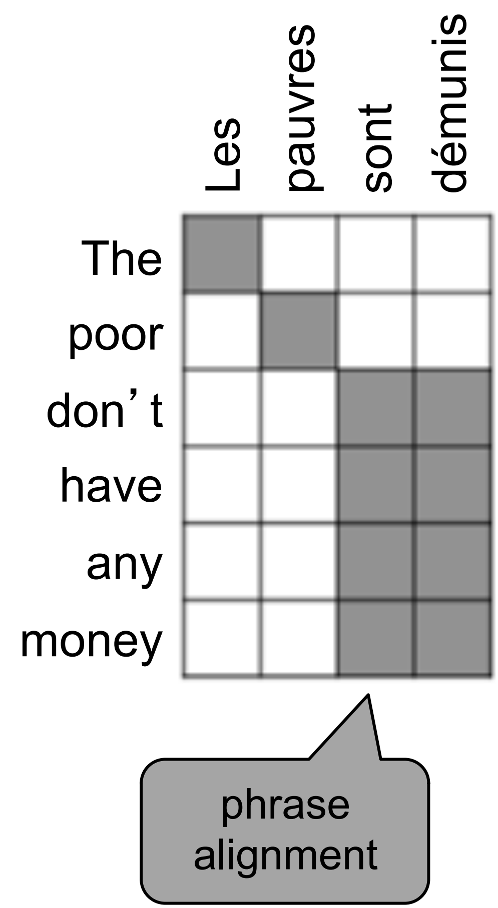




Traditional Machine Translation

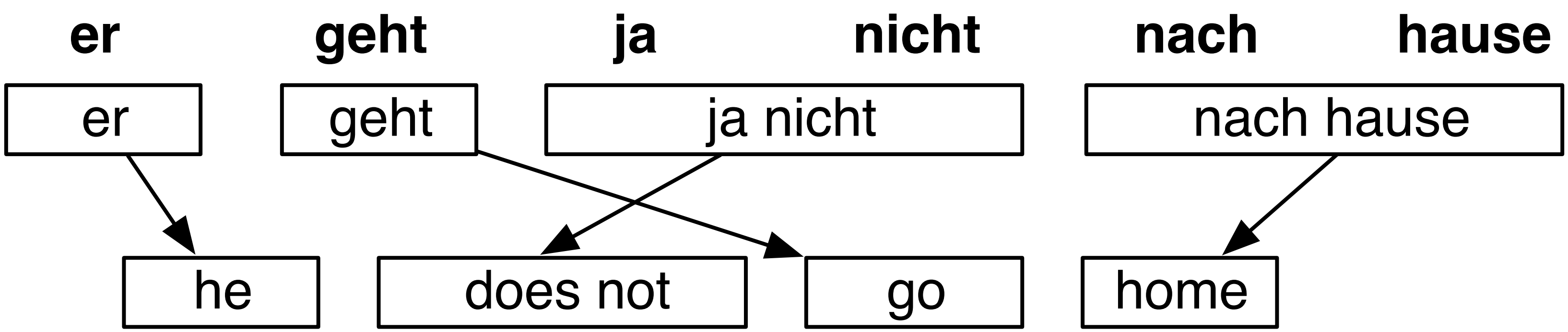
Step 1: Alignment





Traditional Machine Translation

- **Next Step: Reordering Model**

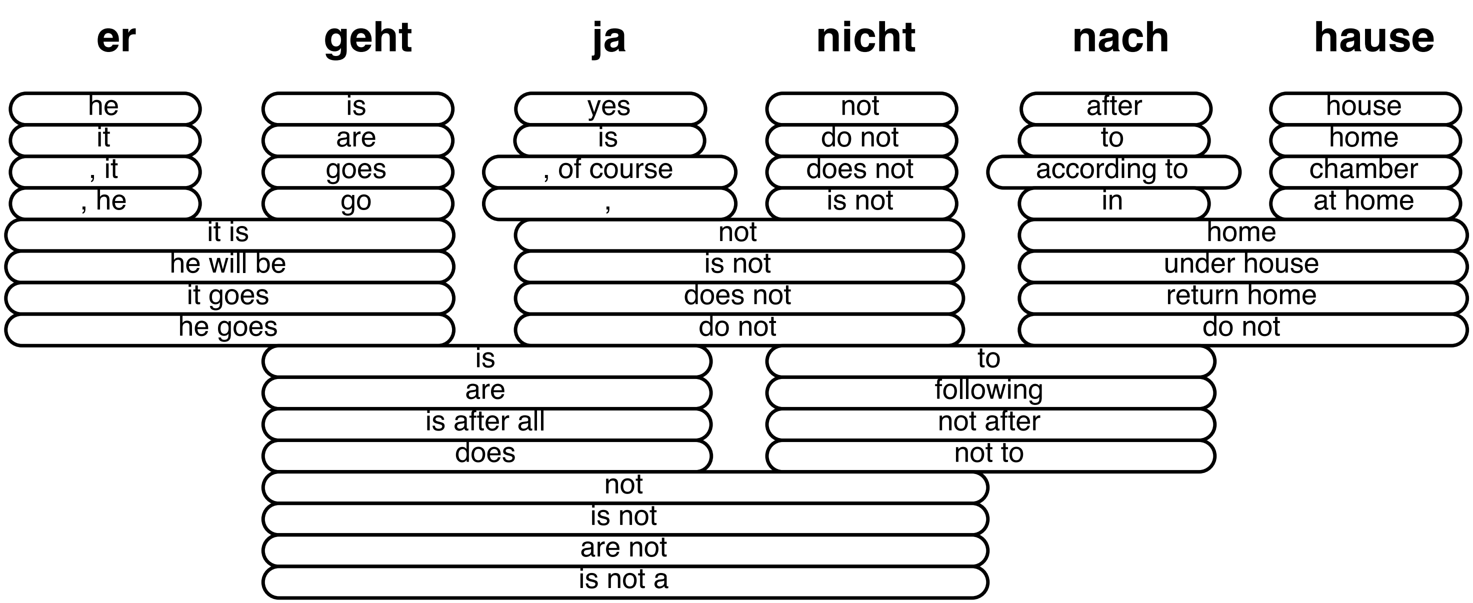


Traditional Machine Translation

- **After many steps…"Phrase Table"**

Each phrase in source language has many possible translations resulting in large search space:

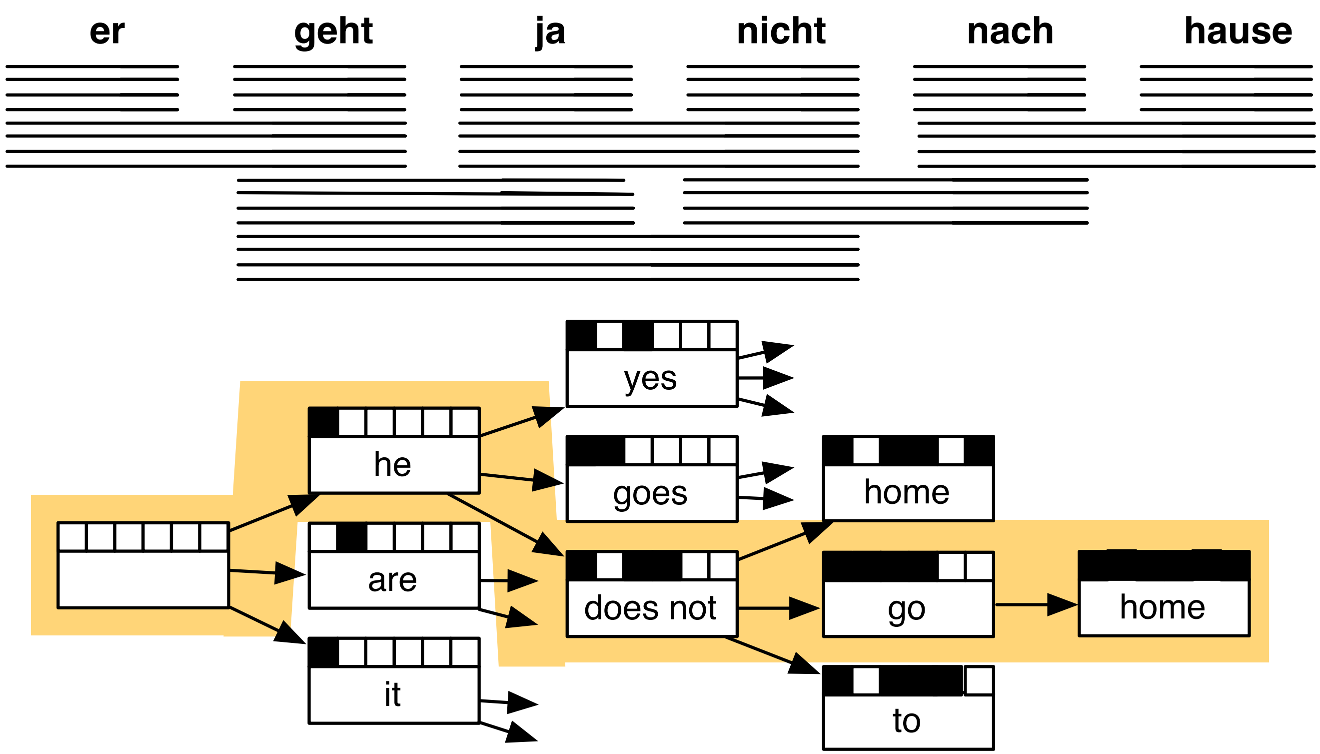
Translation Options



Traditional Machine Translation

Decode: Search for best of many hypotheses

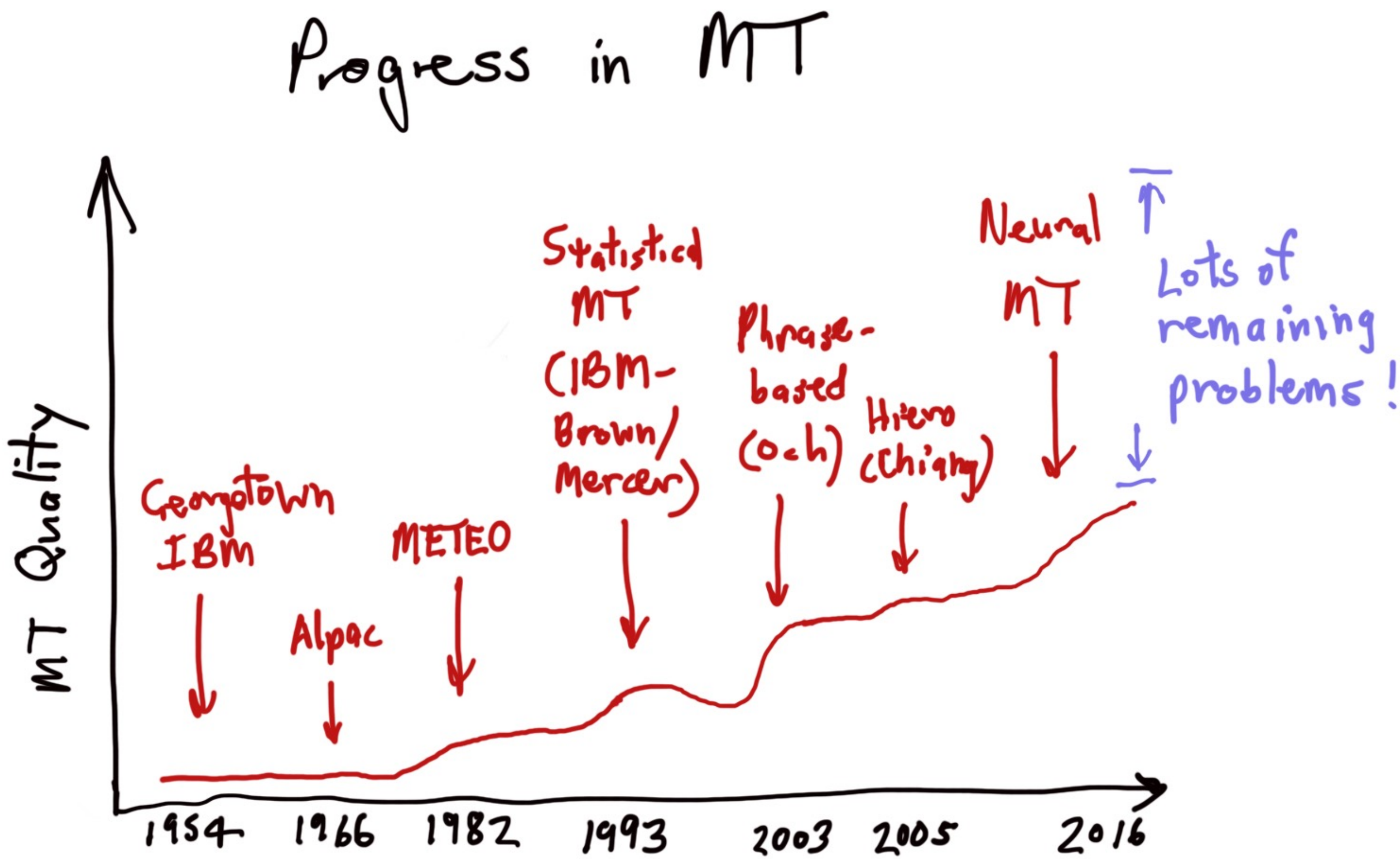
Hard search problem that also includes language model



Traditional Machine Translation

- Lots of feature engineering
- Very complex pipeline systems with multiple steps
- Incentive to do it end to end and jointly
- Can neural models be a powerful enough alternative to do so?

Machine Translation Progress

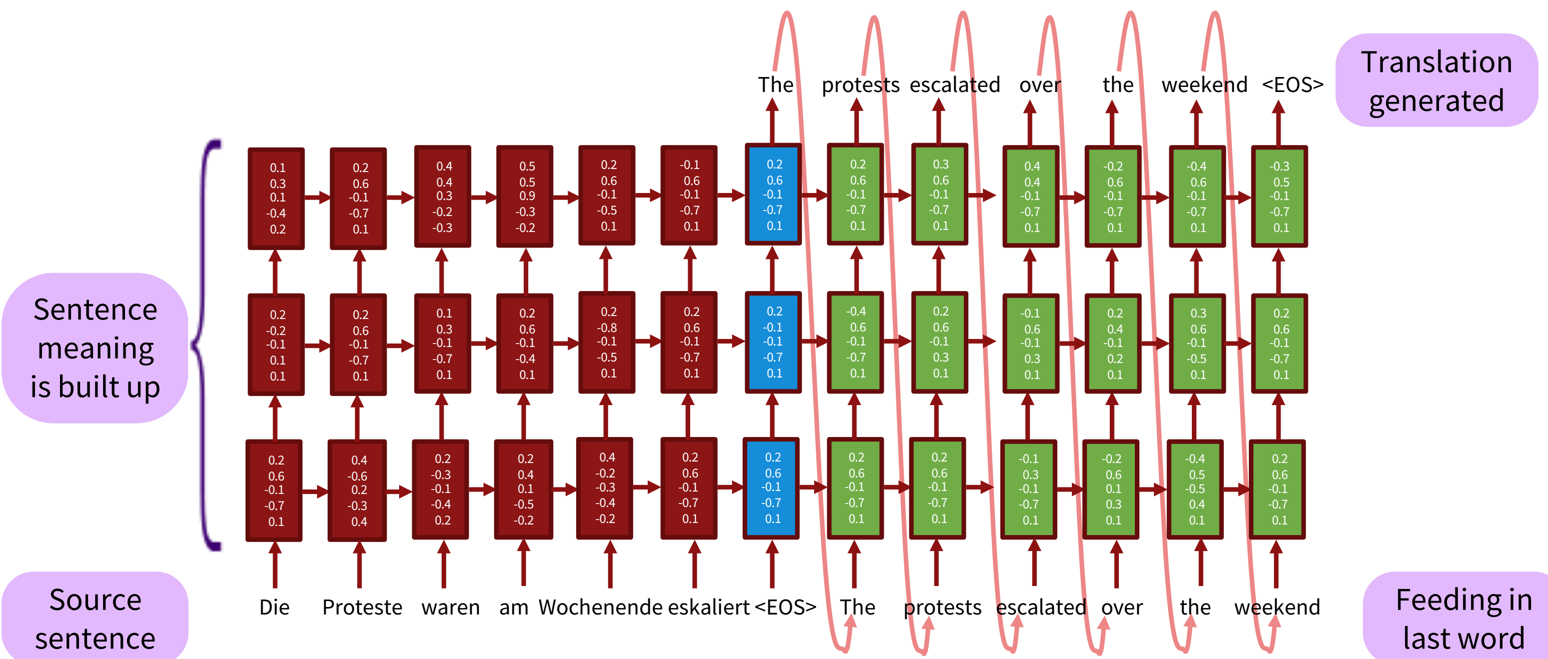


NMT slides from ACL 2016 Tutorial (Luong, Cho, Manning)

Neural Machine Translation

- Encoder-Decoder RNN models:

[Sutskever et al. 2014, Bahdanau et al. 2014, et seq.]
following [Jordan 1986] and more closely [Elman 1990]



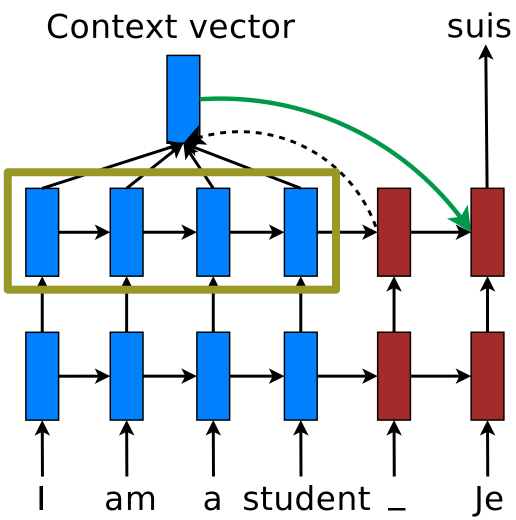
A deep recurrent neural network

Improvement Sources

- Stacking multiple layers
- Bidirectionality
- Better memory units, e.g., GRUs
- Pre-trained language models on tons of monolingual data
- Ensembles
- Attention/Alignment models

Alignment/Attention Models

- Translating longer sentences better, e.g., via attention/alignment module between encoder and decoder to jointly learn alignments and translations end-to-end

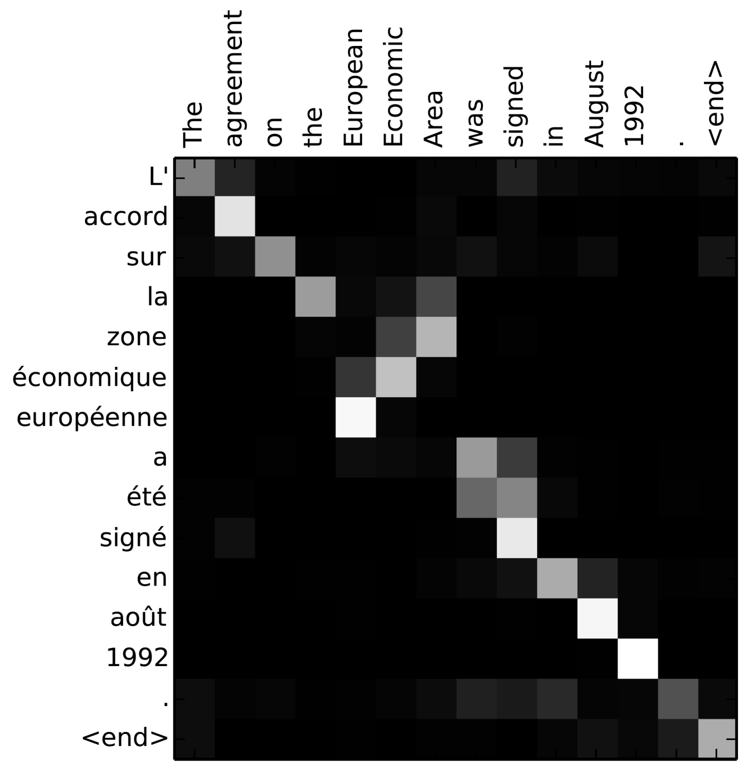


Bilinear form: well-adopted.

$$\mathrm{score}(\boldsymbol{h}_t, \bar{\boldsymbol{h}}_s) = \begin{cases} \boldsymbol{h}_t^\top \bar{\boldsymbol{h}}_s \\ \boldsymbol{h}_t^\top \boldsymbol{W_a} \bar{\boldsymbol{h}}_s \\ \boldsymbol{v}_a^\top \tanh\left(\boldsymbol{W_a}[\boldsymbol{h}_t; \bar{\boldsymbol{h}}_s]\right) \end{cases}$$

Alignment/Attention Models

- Translating longer sentences better, e.g., via attention/alignment module between encoder and decoder to jointly learn alignments and translations end-to-end

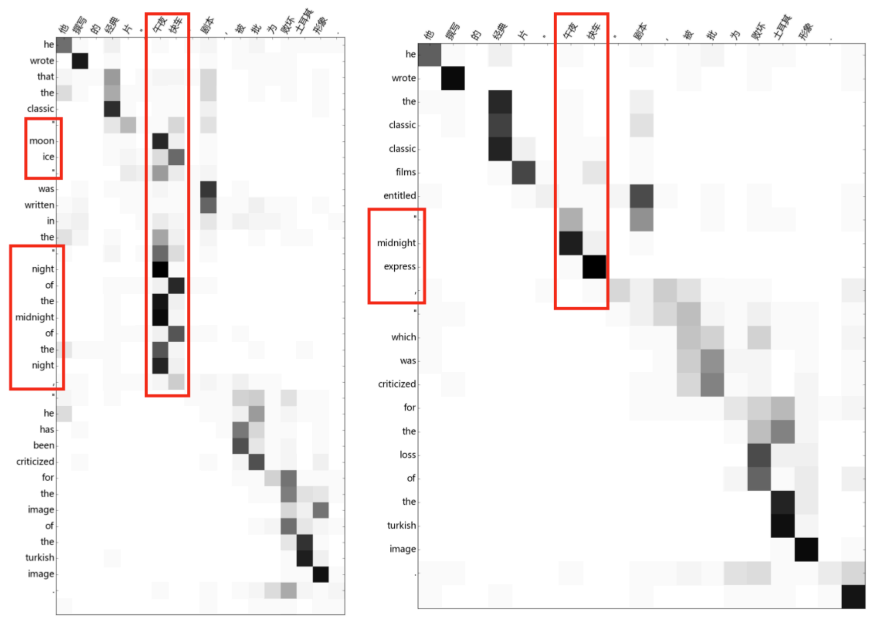


Dzmitry Bahdanau, KyungHuyn Cho, and Yoshua Bengio. ***Neural Machine Translation by Jointly Learning to Translate and Align****. ICLR'15.*

Linguistics Insights

Constraints on “distortion” (displacement) and fertility

➜ **Constraints on attention** [Cohn, Hoang, Vymolova, Yao, Dyer & Haffari NAACL 2016; Feng, Liu, Li, Zhou 2016 arXiv; Yang, Hu, Deng, Dyer, Smola 2016 arXiv].



Linguistics Insights

Extend to NMT – *Linguistic insights*

- [Cohn, Hoang, Vymolova, Yao, Dyer, Haffari, NAACL'16]: position (IBM2) + Markov (HMM) + fertility (IBM3-5) + alignment symmetry (BerkeleyAligner).

$$-\log(P(\mathbf{y}|\mathbf{x})) + \lambda \sum_{i}^{L} (1 - \sum_{t}^{C} \alpha_{ti})^2$$

Per source word   Source word fertility

- [Tu, Lu, Liu, Liu, Li, ACL'16]: linguistic & NN-based coverage models.

Other New Ideas/Improvements

- Extending vocabulary coverage and handling rare/unseen words
- Handling more language variations, e.g., via character-level models to capture morphology
- Utilize more data resources, e.g., multilingual models (one to many, many to one, many to many), multi-task learning (combine with other encoder-decoder tasks with shared sides)
- See ACL 2016 tutorial: https://sites.google.com/site/acl16nmt/

Next Class

- We will start paper reading from next class!
- Summary Writing every week and Discussion Leaders
- Class participation in discussions and brainstorming!
- First topic will be Question Answering
- Please send me top-3 week/topic choices by tomorrow 5pm.
- Will finalize discussion leaders and email everyone