NLP 101

Maury Courtland (www.maury.science)

June 11, 2019

Talk guidelines

Talk guidelines

- Everything is at a fairly high level

Talk guidelines

- Everything is at a fairly high level
  - If you're interested in learning more, please ask

Talk guidelines

- Everything is at a fairly high level
  - If you're interested in learning more, please ask
  - If it's too much to cover this time, we'll table it for later

Talk guidelines

- Everything is at a fairly high level
  - If you're interested in learning more, please ask
  - If it's too much to cover this time, we'll table it for later
- Please ask a question whenever it comes up

Talk guidelines

- Everything is at a fairly high level
  - If you're interested in learning more, please ask
  - If it's too much to cover this time, we'll table it for later
- Please ask a question whenever it comes up
  - Don't call out in the middle, just raise your hand and I'll call on you whenever a good time to break comes up

Talk guidelines

- Everything is at a fairly high level
  - If you're interested in learning more, please ask
  - If it's too much to cover this time, we'll table it for later
- Please ask a question whenever it comes up
  - Don't call out in the middle, just raise your hand and I'll call on you whenever a good time to break comes up
  - I check in with my audience a lot so I should be pretty quick in getting to your question

Talk guidelines

- Everything is at a fairly high level
  - If you're interested in learning more, please ask
  - If it's too much to cover this time, we'll table it for later
- Please ask a question whenever it comes up
  - Don't call out in the middle, just raise your hand and I'll call on you whenever a good time to break comes up
  - I check in with my audience a lot so I should be pretty quick in getting to your question
- If there's something you don't understand, please ask

Talk guidelines

- Everything is at a fairly high level
  - If you're interested in learning more, please ask
  - If it's too much to cover this time, we'll table it for later
- Please ask a question whenever it comes up
  - Don't call out in the middle, just raise your hand and I'll call on you whenever a good time to break comes up
  - I check in with my audience a lot so I should be pretty quick in getting to your question
- If there's something you don't understand, please ask
  - You're probably not the only one

Talk guidelines

- Everything is at a fairly high level
  - If you're interested in learning more, please ask
  - If it's too much to cover this time, we'll table it for later
- Please ask a question whenever it comes up
  - Don't call out in the middle, just raise your hand and I'll call on you whenever a good time to break comes up
  - I check in with my audience a lot so I should be pretty quick in getting to your question
- If there's something you don't understand, please ask
  - You're probably not the only one
  - Even if you are, everyone would benefit from a reframing of the topic

Talk guidelines

- Everything is at a fairly high level
  - If you're interested in learning more, please ask
  - If it's too much to cover this time, we'll table it for later
- Please ask a question whenever it comes up
  - Don't call out in the middle, just raise your hand and I'll call on you whenever a good time to break comes up
  - I check in with my audience a lot so I should be pretty quick in getting to your question
- If there's something you don't understand, please ask
  - You're probably not the only one
  - Even if you are, everyone would benefit from a reframing of the topic
- Image credits in accompanying Markdown file as image alttext

Linguistics/Theoretical Background

Information

- Shannon information/entropy/surprisal

$$H = -\sum_i p_i \log_2 (p_i)$$

Figure 1: Shannon Entropy

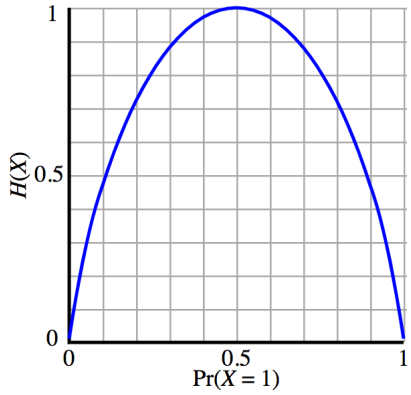


Figure 2: Binomial Entropy Function

A brief history of time

- The “time sandwich” of language

A brief history of time

- The "time sandwich" of language
  - TEMPORAL: segments (phonotactics) and speech (articulatory dynamics, etc.)

A brief history of time

- The "time sandwich" of language
  - TEMPORAL: segments (phonotactics) and speech (articulatory dynamics, etc.)
  - ATEMPORAL: morphemes and words and sentences are not really temporal (evidenced by free variation in word order, etc.)

A brief history of time

- The "time sandwich" of language
  - TEMPORAL: segments (phonotactics) and speech (articulatory dynamics, etc.)
  - ATEMPORAL: morphemes and words and sentences are not really temporal (evidenced by free variation in word order, etc.)
  - TEMPORAL: discourse and dialog are temporal again (no language starts a story with "that's all folks")

A brief history of time

- The "time sandwich" of language
  - TEMPORAL: segments (phonotactics) and speech (articulatory dynamics, etc.)
  - ATEMPORAL: morphemes and words and sentences are not really temporal (evidenced by free variation in word order, etc.)
  - TEMPORAL: discourse and dialog are temporal again (no language starts a story with "that's all folks")
- The validity of bidirectionality is thus questionable at levels that are temporal

A brief history of time

- The "time sandwich" of language
  - TEMPORAL: segments (phonotactics) and speech (articulatory dynamics, etc.)
  - ATEMPORAL: morphemes and words and sentences are not really temporal (evidenced by free variation in word order, etc.)
  - TEMPORAL: discourse and dialog are temporal again (no language starts a story with "that's all folks")
- The validity of bidirectionality is thus questionable at levels that are temporal
  - Patient2vec redux: there's a reason we call people who had cancer but now don't in "remission" rather than "healthy"

A brief history of time

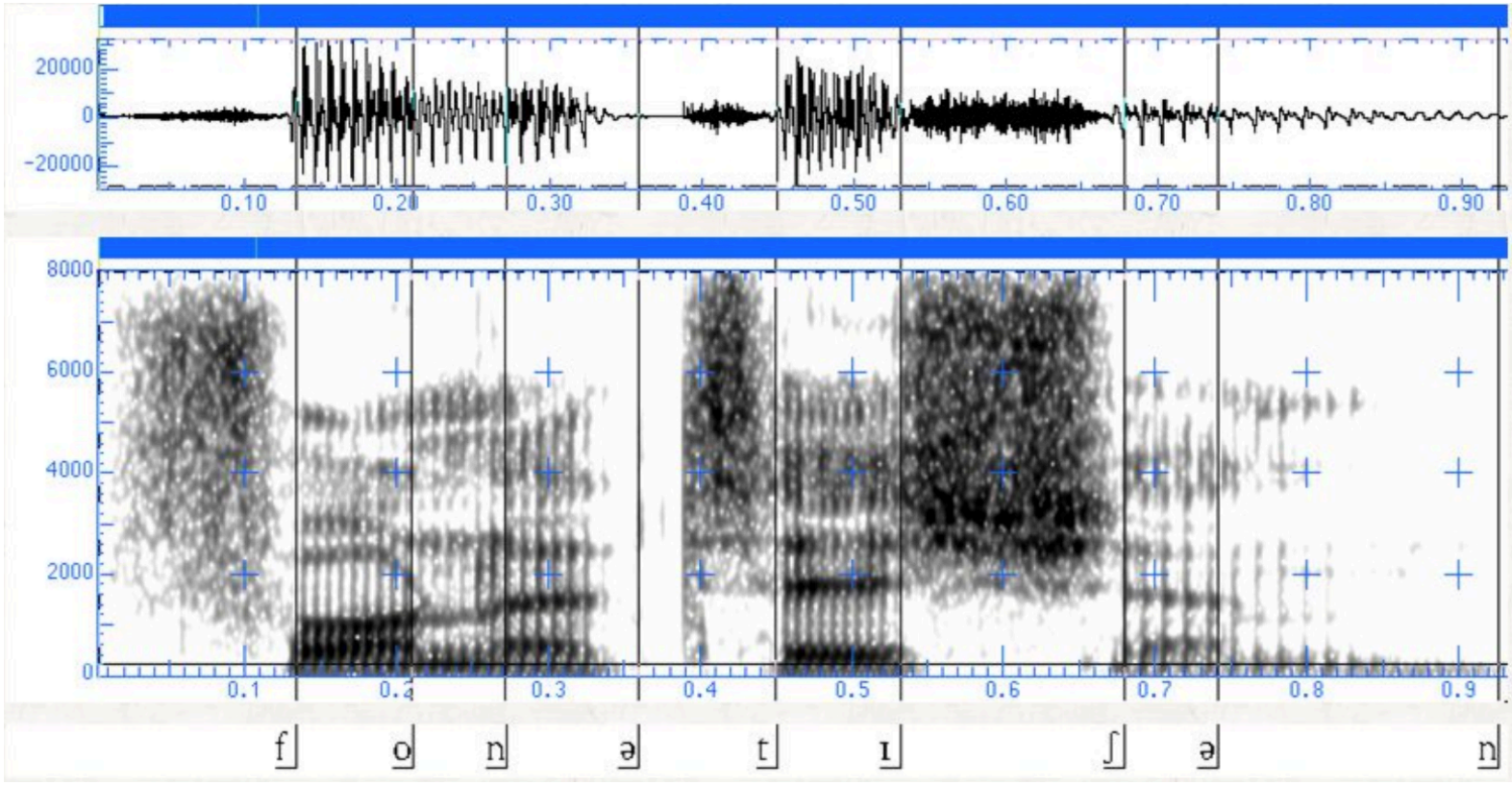
- The "time sandwich" of language
  - TEMPORAL: segments (phonotactics) and speech (articulatory dynamics, etc.)
  - ATEMPORAL: morphemes and words and sentences are not really temporal (evidenced by free variation in word order, etc.)
  - TEMPORAL: discourse and dialog are temporal again (no language starts a story with "that's all folks")
- The validity of bidirectionality is thus questionable at levels that are temporal
  - Patient2vec redux: there's a reason we call people who had cancer but now don't in "remission" rather than "healthy"
  - BUT if you don't yet have cancer, we don't say you're in "premission"

A brief history of time

- The "time sandwich" of language
  - TEMPORAL: segments (phonotactics) and speech (articulatory dynamics, etc.)
  - ATEMPORAL: morphemes and words and sentences are not really temporal (evidenced by free variation in word order, etc.)
  - TEMPORAL: discourse and dialog are temporal again (no language starts a story with "that's all folks")
- The validity of bidirectionality is thus questionable at levels that are temporal
  - Patient2vec redux: there's a reason we call people who had cancer but now don't in "remission" rather than "healthy"
  - BUT if you don't yet have cancer, we don't say you're in "premission"
  - Time is always marching forward so you have to be careful about the implications of modeling choices of uni-directional vs. bi-directional

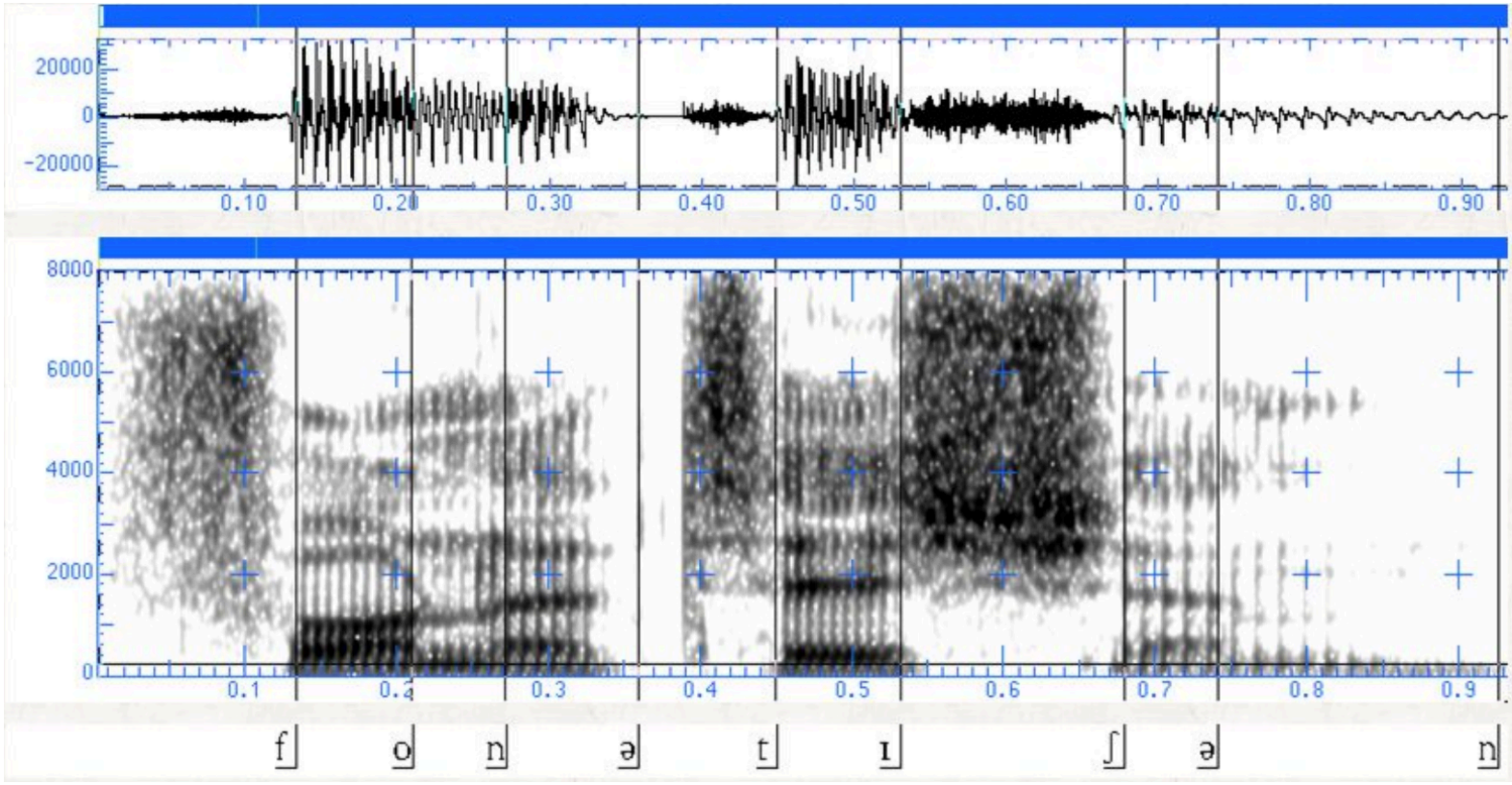
Speech (TEMPORAL) vs. Language (ATEMPORAL)

- Sublexical sound units



Speech (TEMPORAL) vs. Language (ATEMPORAL)

- Sublexical sound units



- Sublexical orthographic units = letters

Speech vs. Language: Differences

- Variance in acoustics/articulation and invariance in orthography (except handwriting)

$$\operatorname{sinc}(x) = \frac{\sin(\pi x)}{\pi x} \,.$$

Figure 3: Sinc Equation

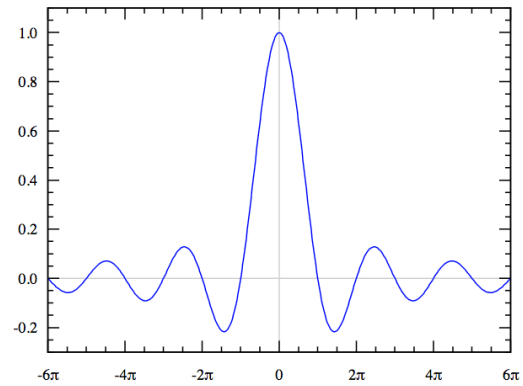


Figure 4: Sinc Function

Speech vs. Language: Differences

- Variance in acoustics/articulation and invariance in orthography (except handwriting)
  - Sounds change by random and by rules

$$\operatorname{sinc}(x) = \frac{\sin(\pi x)}{\pi x} \,.$$

Figure 3: Sinc Equation

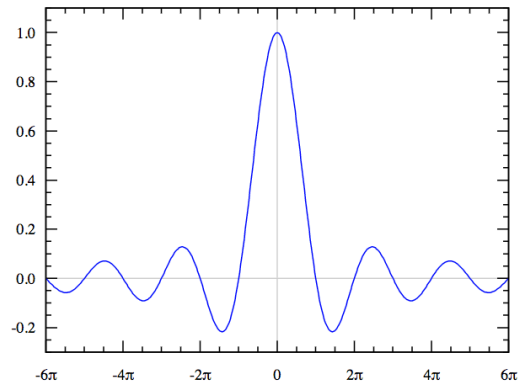


Figure 4: Sinc Function

Speech vs. Language: Differences

- Variance in acoustics/articulation and invariance in orthography (except handwriting)
  - Sounds change by random and by rules
- Discrete time steps vs. continuous time steps (i.e. one is imperfectly sampled, the other perfectly captured)

$$\operatorname{sinc}(x) = \frac{\sin(\pi x)}{\pi x} \, .$$

Figure 3: Sinc Equation

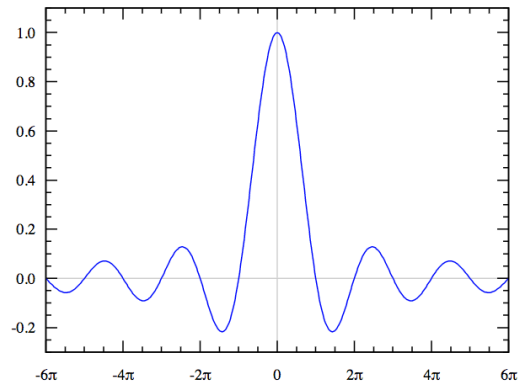


Figure 4: Sinc Function

Speech vs. Language: Differences

- Variance in acoustics/articulation and invariance in orthography (except handwriting)
  - Sounds change by random and by rules
- Discrete time steps vs. continuous time steps (i.e. one is imperfectly sampled, the other perfectly captured)
  - Speech sampling, the sinc function, perfect reconstruction (with max fs/2)

$$\operatorname{sinc}(x) = \frac{\sin(\pi x)}{\pi x} \,.$$

Figure 3: Sinc Equation

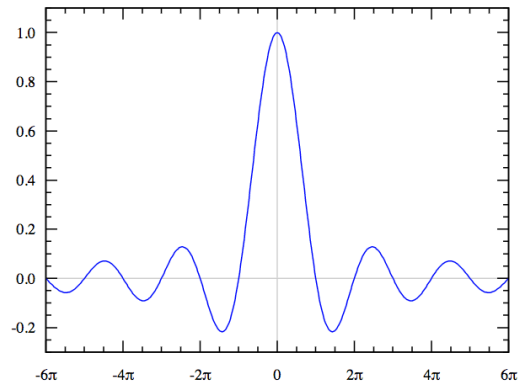
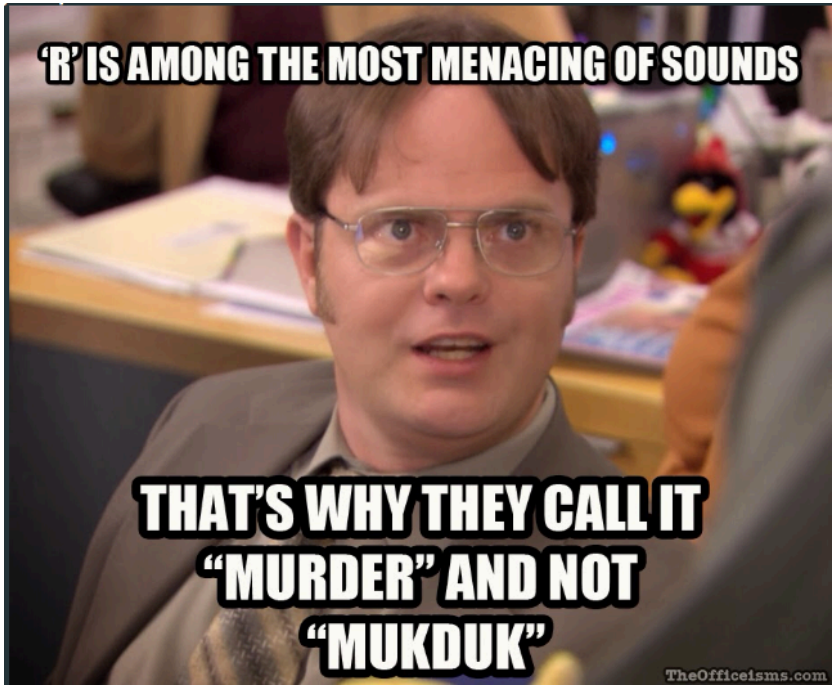


Figure 4: Sinc Function

Speech vs. Language: Similarities

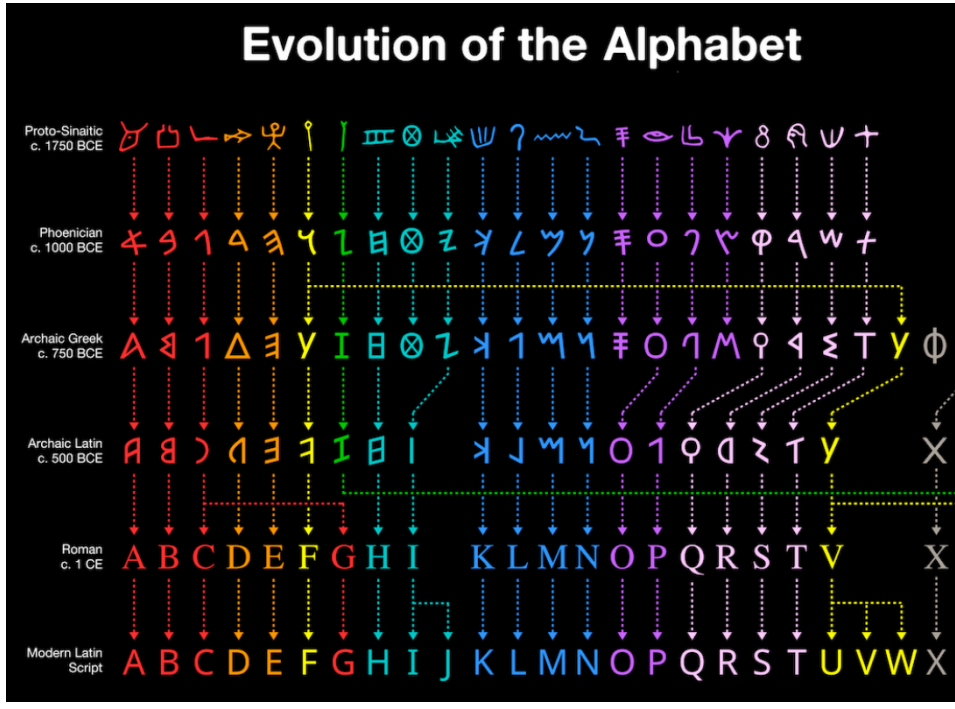
- Both are arbitrary:

Sounds:



(minus onomatopoeias)

Letters:



Meaning et al. (ATEMPORAL)

- Syntax: word order freely varies across languages (SVO, OVS, SOV, etc. have same meaning)

Meaning et al. (ATEMPORAL)

- Syntax: word order freely varies across languages (SVO, OVS, SOV, etc. have same meaning)
- (Distributional) Semantics: locality matters

Meaning et al. (ATEMPORAL)

- Syntax: word order freely varies across languages (SVO, OVS, SOV, etc. have same meaning)
- (Distributional) Semantics: locality matters
  - "You shall know a word by the company it keeps" - Firth, 1957

Meaning et al. (ATEMPORAL)

- Syntax: word order freely varies across languages (SVO, OVS, SOV, etc. have same meaning)
- (Distributional) Semantics: locality matters
  - "You shall know a word by the company it keeps" - Firth, 1957
- Hierarchical vs. Linear (Spoiler: stack LSTM marries the 2)

Meaning et al. (ATEMPORAL)

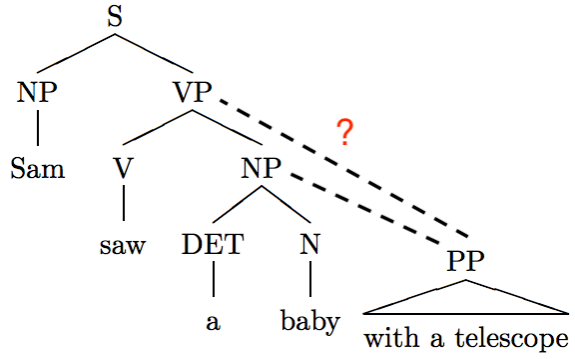
- Syntax: word order freely varies across languages (SVO, OVS, SOV, etc. have same meaning)
- (Distributional) Semantics: locality matters
  - "You shall know a word by the company it keeps" - Firth, 1957
- Hierarchical vs. Linear (Spoiler: stack LSTM marries the 2)
  - Language is necessarily constrained to linearity given directionality in orthography and speech acts

Meaning et al. (ATEMPORAL)

- Syntax: word order freely varies across languages (SVO, OVS, SOV, etc. have same meaning)
- (Distributional) Semantics: locality matters
  - "You shall know a word by the company it keeps" - Firth, 1957
- Hierarchical vs. Linear (Spoiler: stack LSTM marries the 2)
  - Language is necessarily constrained to linearity given directionality in orthography and speech acts
  - But simultaneously has a rich hierarchical structure not necessarily obvious from the surface linear form

Meaning et al. (ATEMPORAL)

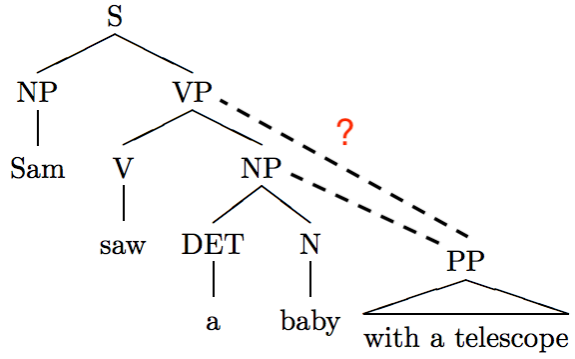
- Syntax: word order freely varies across languages (SVO, OVS, SOV, etc. have same meaning)
- (Distributional) Semantics: locality matters
  - "You shall know a word by the company it keeps" - Firth, 1957
- Hierarchical vs. Linear (Spoiler: stack LSTM marries the 2)
  - Language is necessarily constrained to linearity given directionality in orthography and speech acts
  - But simultaneously has a rich hierarchical structure not necessarily obvious from the surface linear form
  - The hierarchical structure can change meaning even when



linear order stays the same:

Meaning et al. (ATEMPORAL)

- Syntax: word order freely varies across languages (SVO, OVS, SOV, etc. have same meaning)
- (Distributional) Semantics: locality matters
  - "You shall know a word by the company it keeps" - Firth, 1957
- Hierarchical vs. Linear (Spoiler: stack LSTM marries the 2)
  - Language is necessarily constrained to linearity given directionality in orthography and speech acts
  - But simultaneously has a rich hierarchical structure not necessarily obvious from the surface linear form
  - The hierarchical structure can change meaning even when linear order stays the same:



- Meaning is not really constrained by time, this makes things like Bidirectional LSTMs kosher

Discourse and dialog (TEMPORAL)

- Propositions: a statement made that can be evaluated with a truth value (boolean logic)

Discourse and dialog (TEMPORAL)

- Propositions: a statement made that can be evaluated with a truth value (boolean logic)
- What happens when statements are not made in a vaccuum (interlocutor mental modeling of common ground, accommodation, etc.)?

Discourse and dialog (TEMPORAL)

- Propositions: a statement made that can be evaluated with a truth value (boolean logic)
- What happens when statements are not made in a vaccuum (interlocutor mental modeling of common ground, accommodation, etc.)?
- Gricean maxims:

Discourse and dialog (TEMPORAL)

- Propositions: a statement made that can be evaluated with a truth value (boolean logic)
- What happens when statements are not made in a vaccuum (interlocutor mental modeling of common ground, accommodation, etc.)?
- Gricean maxims:
  1. Quantity: try to be as informative as one possibly can, and give as much information as is needed, and no more.

Discourse and dialog (TEMPORAL)

- Propositions: a statement made that can be evaluated with a truth value (boolean logic)
- What happens when statements are not made in a vaccuum (interlocutor mental modeling of common ground, accommodation, etc.)?
- Gricean maxims:
  1. Quantity: try to be as informative as one possibly can, and give as much information as is needed, and no more.
  2. Quality: try to be truthful, and not give information that is false or that is not supported by evidence.

Discourse and dialog (TEMPORAL)

- Propositions: a statement made that can be evaluated with a truth value (boolean logic)
- What happens when statements are not made in a vaccuum (interlocutor mental modeling of common ground, accommodation, etc.)?
- Gricean maxims:
  1. Quantity: try to be as informative as one possibly can, and give as much information as is needed, and no more.
  2. Quality: try to be truthful, and not give information that is false or that is not supported by evidence.
  3. Relation: try to be relevant, and says things that are pertinent to the discussion.

Discourse and dialog (TEMPORAL)

- Propositions: a statement made that can be evaluated with a truth value (boolean logic)
- What happens when statements are not made in a vaccuum (interlocutor mental modeling of common ground, accommodation, etc.)?
- Gricean maxims:
  1. Quantity: try to be as informative as one possibly can, and give as much information as is needed, and no more.
  2. Quality: try to be truthful, and not give information that is false or that is not supported by evidence.
  3. Relation: try to be relevant, and says things that are pertinent to the discussion.
  4. Manner: try to be as clear, as brief, and as orderly as you can in what one says, and avoid obscurity and ambiguity.

Discourse and dialog (TEMPORAL)

- Propositions: a statement made that can be evaluated with a truth value (boolean logic)
- What happens when statements are not made in a vaccuum (interlocutor mental modeling of common ground, accommodation, etc.)?
- Gricean maxims:
  1. Quantity: try to be as informative as one possibly can, and give as much information as is needed, and no more.
  2. Quality: try to be truthful, and not give information that is false or that is not supported by evidence.
  3. Relation: try to be relevant, and says things that are pertinent to the discussion.
  4. Manner: try to be as clear, as brief, and as orderly as you can in what one says, and avoid obscurity and ambiguity.
- How do you establish a common ground of exchange as a conversation group? (diapix, etc.)

Discourse and dialog (TEMPORAL)

- Propositions: a statement made that can be evaluated with a truth value (boolean logic)
- What happens when statements are not made in a vaccuum (interlocutor mental modeling of common ground, accommodation, etc.)?
- Gricean maxims:
  1. Quantity: try to be as informative as one possibly can, and give as much information as is needed, and no more.
  2. Quality: try to be truthful, and not give information that is false or that is not supported by evidence.
  3. Relation: try to be relevant, and says things that are pertinent to the discussion.
  4. Manner: try to be as clear, as brief, and as orderly as you can in what one says, and avoid obscurity and ambiguity.
- How do you establish a common ground of exchange as a conversation group? (diapix, etc.)
- How does dialog (alternatively, narrative) evolve over time?

Design questions

How do we obtain continuous embeddings from discrete entities?

- Letters: how can we tell that vowels are more similar to each other than consonants (their ascii codes/vocabulary instances are categorical NOT ordinal)

How do we obtain continuous embeddings from discrete entities?

- Letters: how can we tell that vowels are more similar to each other than consonants (their ascii codes/vocabulary instances are categorical NOT ordinal)
- Words: how can we tell that some words are more related than others either syntactically (i.e. parts of speech) or semantically (i.e. thematically related) rather than just having a one-hot for the vocabulary word that is being encoded

How do we obtain continuous embeddings from discrete entities?

- Letters: how can we tell that vowels are more similar to each other than consonants (their ascii codes/vocabulary instances are categorical NOT ordinal)
- Words: how can we tell that some words are more related than others either syntactically (i.e. parts of speech) or semantically (i.e. thematically related) rather than just having a one-hot for the vocabulary word that is being encoded
- Phrase/sentences: theoretically infinite possible combinations of words, which while at the surface differ but semantically are very similar

How do we obtain continuous embeddings from discrete entities?

- Letters: how can we tell that vowels are more similar to each other than consonants (their ascii codes/vocabulary instances are categorical NOT ordinal)
- Words: how can we tell that some words are more related than others either syntactically (i.e. parts of speech) or semantically (i.e. thematically related) rather than just having a one-hot for the vocabulary word that is being encoded
- Phrase/sentences: theoretically infinite possible combinations of words, which while at the surface differ but semantically are very similar
  - "I sipped a cup of joe" V.

How do we obtain continuous embeddings from discrete entities?

- Letters: how can we tell that vowels are more similar to each other than consonants (their ascii codes/vocabulary instances are categorical NOT ordinal)
- Words: how can we tell that some words are more related than others either syntactically (i.e. parts of speech) or semantically (i.e. thematically related) rather than just having a one-hot for the vocabulary word that is being encoded
- Phrase/sentences: theoretically infinite possible combinations of words, which while at the surface differ but semantically are very similar
  - "I sipped a cup of joe" V.
  - "He drank his mug of coffee"

How do we obtain continuous embeddings from discrete entities?

- Letters: how can we tell that vowels are more similar to each other than consonants (their ascii codes/vocabulary instances are categorical NOT ordinal)
- Words: how can we tell that some words are more related than others either syntactically (i.e. parts of speech) or semantically (i.e. thematically related) rather than just having a one-hot for the vocabulary word that is being encoded
- Phrase/sentences: theoretically infinite possible combinations of words, which while at the surface differ but semantically are very similar
  - "I sipped a cup of joe" V.
  - "He drank his mug of coffee"
- Documents: definitely infinite possibilities, but we still need to be able to cluster them based on sentiment, topic, etc. to be useful to extend to out of training instances (which will be almost all documents)

What level of structure do you model text at?

- Letter/Subword?: fasttext (https://research.fb.com/fasttext/)

What level of structure do you model text at?

- Letter/Subword?: fasttext (https://research.fb.com/fasttext/)
- Word?: Word2Vec (https://pypi.org/project/word2vec/)

What level of structure do you model text at?

- Letter/Subword?: fasttext (https://research.fb.com/fasttext/)
- Word?: Word2Vec (https://pypi.org/project/word2vec/)
- Phrase (a.k.a. constituent, e.g. noun phrase)?: RNTN (https://github.com/alsoltani/RNTN)

What level of structure do you model text at?

- Letter/Subword?: fasttext (https://research.fb.com/fasttext/)
- Word?: Word2Vec (https://pypi.org/project/word2vec/)
- Phrase (a.k.a. constituent, e.g. noun phrase)?: RNTN (https://github.com/alsoltani/RNTN)
- Sentence?: Skip-thought vectors (https://github.com/ryankiros/skip-thoughts)

What level of structure do you model text at?

- Letter/Subword?: fasttext (https://research.fb.com/fasttext/)
- Word?: Word2Vec (https://pypi.org/project/word2vec/)
- Phrase (a.k.a. constituent, e.g. noun phrase)?: RNTN (https://github.com/alsoltani/RNTN)
- Sentence?: Skip-thought vectors (https://github.com/ryankiros/skip-thoughts)
- Document?: TF-IDF, LDA, etc.

(Some) common problems and why we care

Machine translation

- Considering the 7k lanugages of the world (or at least the ~50 represented on the internet), it would be nice to not have language be a barrier to human exchange (shown to create insular web communities)

Machine translation

- Considering the 7k lanugages of the world (or at least the ~50 represented on the internet), it would be nice to not have language be a barrier to human exchange (shown to create insular web communities)
  - Google translate

Machine translation

- Considering the 7k lanugages of the world (or at least the ~50 represented on the internet), it would be nice to not have language be a barrier to human exchange (shown to create insular web communities)
  - Google translate
  - Other boutique translation sites

Machine translation

- Considering the 7k lanugages of the world (or at least the ~50 represented on the internet), it would be nice to not have language be a barrier to human exchange (shown to create insular web communities)
  - Google translate
  - Other boutique translation sites
  - Legal translations

Machine translation

- Considering the 7k lanugages of the world (or at least the ~50 represented on the internet), it would be nice to not have language be a barrier to human exchange (shown to create insular web communities)
  - Google translate
  - Other boutique translation sites
  - Legal translations
  - Medical translations

Sentiment analysis and emotion detection

- Given a language sample, we would like to detect how the speaker/writer feels about the topic they're under discussion (either well defined or estimated by topic modeling)

Sentiment analysis and emotion detection

- Given a language sample, we would like to detect how the speaker/writer feels about the topic they're under discussion (either well defined or estimated by topic modeling)
  - Product reviews (Amazon, Ebay, Yelp, Goodreads, etc.)

Sentiment analysis and emotion detection

- Given a language sample, we would like to detect how the speaker/writer feels about the topic they're under discussion (either well defined or estimated by topic modeling)
  - Product reviews (Amazon, Ebay, Yelp, Goodreads, etc.)
  - Chatbots (to react to user emotion: XiaoIce)

Sentiment analysis and emotion detection

- Given a language sample, we would like to detect how the speaker/writer feels about the topic they're under discussion (either well defined or estimated by topic modeling)
  - Product reviews (Amazon, Ebay, Yelp, Goodreads, etc.)
  - Chatbots (to react to user emotion: XiaoIce)
  - Reddit?

Sentiment analysis and emotion detection

- Given a language sample, we would like to detect how the speaker/writer feels about the topic they're under discussion (either well defined or estimated by topic modeling)
  - Product reviews (Amazon, Ebay, Yelp, Goodreads, etc.)
  - Chatbots (to react to user emotion: XiaoIce)
  - Reddit?
  - Suicide prevention on online forums/ from online presences

Grammar parsing, grammar checking, POS (part of speech) tagging

- Given a language sample, we would like to be able to label all the words their correct parts of speech and build a syntactic model of the sentence

Grammar parsing, grammar checking, POS (part of speech) tagging

- Given a language sample, we would like to be able to label all the words their correct parts of speech and build a syntactic model of the sentence
  - Aids in numerous downstream tasks

Grammar parsing, grammar checking, POS (part of speech) tagging

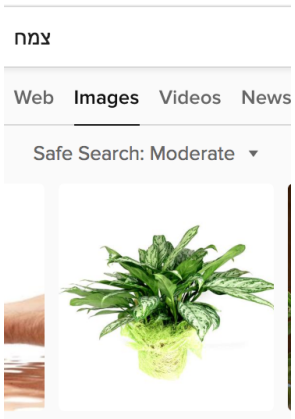
- Given a language sample, we would like to be able to label all the words their correct parts of speech and build a syntactic model of the sentence
  - Aids in numerous downstream tasks
  - Grammar checking/advice (Grammarly, Microsoft Word)

Grammar parsing, grammar checking, POS (part of speech) tagging

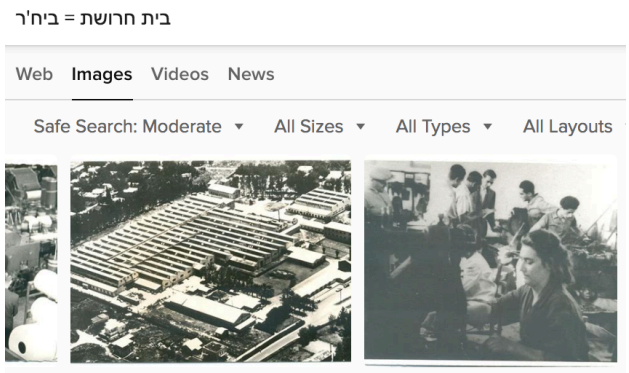
- Given a language sample, we would like to be able to label all the words their correct parts of speech and build a syntactic model of the sentence
  - Aids in numerous downstream tasks
  - Grammar checking/advice (Grammarly, Microsoft Word)
  - Automatic role labeling (Subject, Object, etc.)

Word sense disambiguation

- Given a homographic word (or homophonic if speech), which of the different meanings does the user wish to convey?

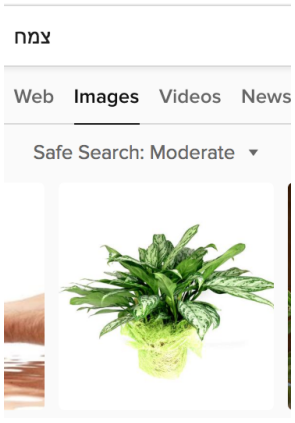


VS.

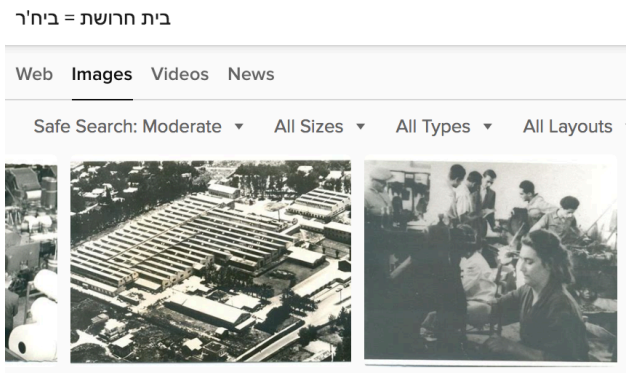


Word sense disambiguation

- Given a homographic word (or homophonic if speech), which of the different meanings does the user wish to convey?
  - Image search (google, yahoo, etc.)

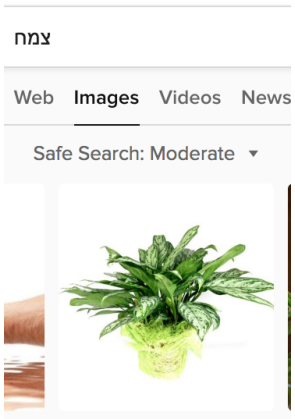


VS.

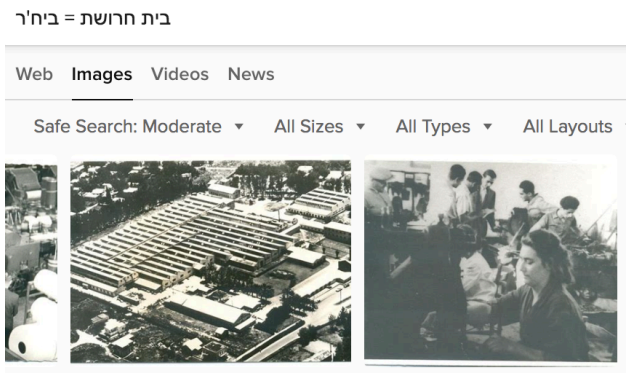


Word sense disambiguation

- Given a homographic word (or homophonic if speech), which of the different meanings does the user wish to convey?
  - Image search (google, yahoo, etc.)
  - Machine translation (the same homograph may be translated differently, e.g. plant:

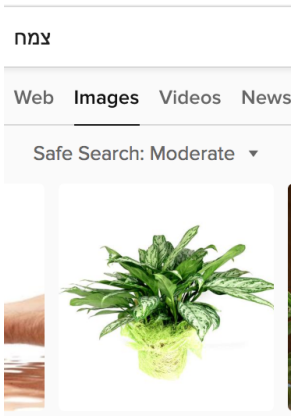


VS.

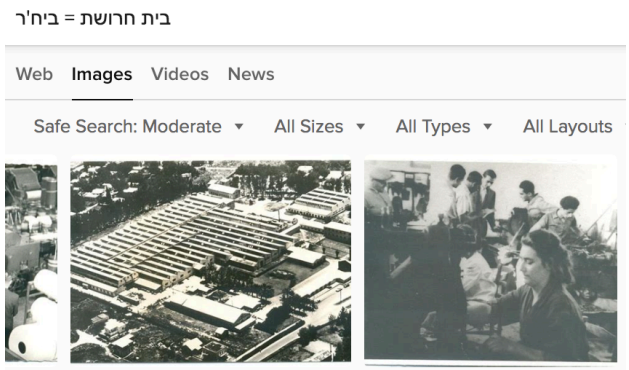


Word sense disambiguation

- Given a homographic word (or homophonic if speech), which of the different meanings does the user wish to convey?
  - Image search (google, yahoo, etc.)
  - Machine translation (the same homograph may be translated differently, e.g. plant:



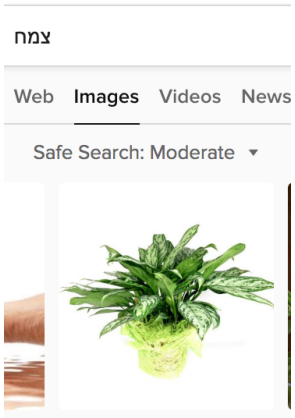
VS.



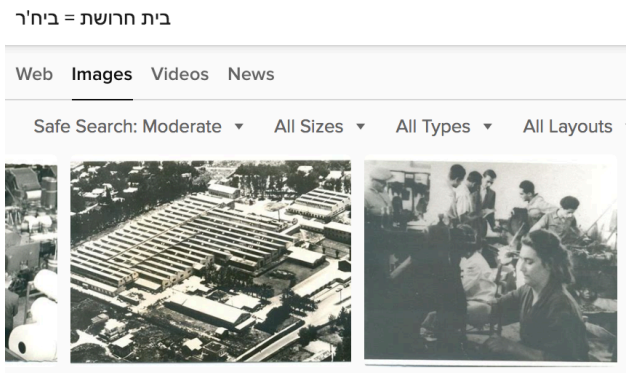
  - Robotics (to navigate or retrieve objects correctly)

Word sense disambiguation

- Given a homographic word (or homophonic if speech), which of the different meanings does the user wish to convey?
  - Image search (google, yahoo, etc.)
  - Machine translation (the same homograph may be translated differently, e.g. plant:



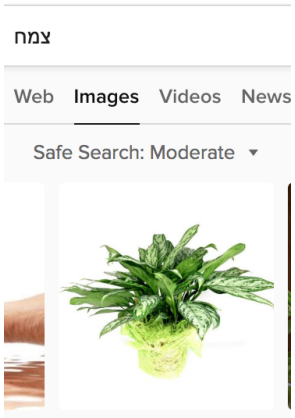
VS.



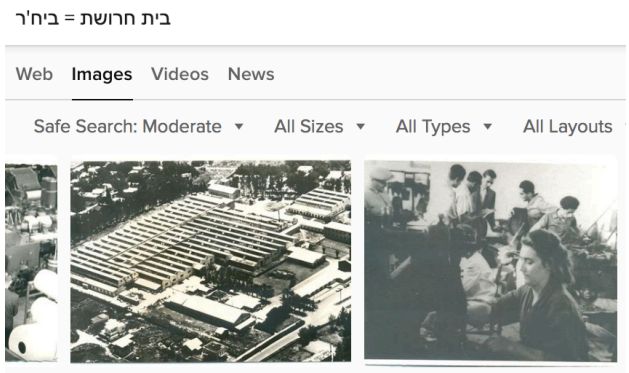
  - Robotics (to navigate or retrieve objects correctly)
  - Conversation agents (to be able to glean meaning and respond sensically)

Word sense disambiguation

- Given a homographic word (or homophonic if speech), which of the different meanings does the user wish to convey?
  - Image search (google, yahoo, etc.)
  - Machine translation (the same homograph may be translated differently, e.g. plant:



VS.



  - Robotics (to navigate or retrieve objects correctly)
  - Conversation agents (to be able to glean meaning and respond sensically)
  - etc.

Traditional approaches

Background

- You will notice that many of these approaches leverage (big for the time) data and statistical analyses to try to solve the problem

Background

- You will notice that many of these approaches leverage (big for the time) data and statistical analyses to try to solve the problem
- Useful to know as many are still used as baseline interpretable approaches when reporting SOTA findings (you should know what they're comparing against as a baseline)

Background

- You will notice that many of these approaches leverage (big for the time) data and statistical analyses to try to solve the problem
- Useful to know as many are still used as baseline interpretable approaches when reporting SOTA findings (you should know what they're comparing against as a baseline)
- It's worth recognizing that the approach of training statistical models on big corpora is fairly old

Background

- You will notice that many of these approaches leverage (big for the time) data and statistical analyses to try to solve the problem
- Useful to know as many are still used as baseline interpretable approaches when reporting SOTA findings (you should know what they're comparing against as a baseline)
- It's worth recognizing that the approach of training statistical models on big corpora is fairly old
- The statistical modeling methods of today are definitely better (even the "traditional" methods are still improving) but the fundamental approach has remained the same

Background

- You will notice that many of these approaches leverage (big for the time) data and statistical analyses to try to solve the problem
- Useful to know as many are still used as baseline interpretable approaches when reporting SOTA findings (you should know what they're comparing against as a baseline)
- It's worth recognizing that the approach of training statistical models on big corpora is fairly old
- The statistical modeling methods of today are definitely better (even the "traditional" methods are still improving) but the fundamental approach has remained the same
- Whether you believe ML is "just statistics" or not, it certainly stems from stats, leverages its history, and borrows heavily from the discipline so while the ML approach is novel, it is not necessarily categorically different from old approaches

Machine translation

History

- Large interest post-WW2 as the success of decyphering codes by machines was a proven concept

History

- Large interest post-WW2 as the success of decyphering codes by machines was a proven concept
- Initial approaches treated translation as a code to be broken (like the “enigma”), but that’s not really how language works (e.g. Native American “Wind Talkers” in the war talking safely over tapped lines)

History

- Large interest post-WW2 as the success of decyphering codes by machines was a proven concept
- Initial approaches treated translation as a code to be broken (like the "enigma"), but that's not really how language works (e.g. Native American "Wind Talkers" in the war talking safely over tapped lines)
  - Also requires massive amount of human oversight to develop the grammatical rules

History

- Large interest post-WW2 as the success of decyphering codes by machines was a proven concept
- Initial approaches treated translation as a code to be broken (like the "enigma"), but that's not really how language works (e.g. Native American "Wind Talkers" in the war talking safely over tapped lines)
  - Also requires massive amount of human oversight to develop the grammatical rules
- Abandoned in the 60s due to overhype (sound like an AI story you've heard of?)

History

- Large interest post-WW2 as the success of decyphering codes by machines was a proven concept
- Initial approaches treated translation as a code to be broken (like the "enigma"), but that's not really how language works (e.g. Native American "Wind Talkers" in the war talking safely over tapped lines)
  - Also requires massive amount of human oversight to develop the grammatical rules
- Abandoned in the 60s due to overhype (sound like an AI story you've heard of?)
- Then statistical methods took the scene and with the ever growing amount of data (particularly parallel corpora, e.g. EUROPARL (EU), Hansard (CA)) took off

Statistical machine translation (SMT)

- Uses both a language model $P(y)$ and translation model $P(y|x)$ models the probability that y is a good translation for x

Statistical machine translation (SMT)

- Uses both a language model $P(y)$ and translation model $P(y|x)$ models the probability that y is a good translation for x

$$Pr(y|x) = \frac{Pr(y)Pr(x|y)}{Pr(x)}.$$

Statistical machine translation (SMT)

- Uses both a language model $P(y)$ and translation model $P(y|x)$ models the probability that y is a good translation for x

$$Pr(y|x) = \frac{Pr(y)Pr(x|y)}{Pr(x)}.$$

$$\hat{y} = \arg\max_{y} Pr(x|y)Pr(y)$$

Statistical machine translation (SMT)

- Uses both a language model $P(y)$ and translation model $P(y|x)$ models the probability that y is a good translation for x
  - $$Pr(y|x) = \frac{Pr(y)Pr(x|y)}{Pr(x)}.$$
  - $$\hat{y} = \arg\max_y Pr(x|y)Pr(y)$$
- Language models (LMs) estimate the probability of an utterance $P(y) = P(w_1...w_n)$ in a target language based on a training corpus

Statistical machine translation (SMT)

- Uses both a language model $P(y)$ and translation model $P(y|x)$ models the probability that y is a good translation for x
  - $$Pr(y|x) = \frac{Pr(y)Pr(x|y)}{Pr(x)}.$$
  - $$\hat{y} = \arg\max_y Pr(x|y)Pr(y)$$
- Language models (LMs) estimate the probability of an utterance $P(y) = P(w_1...w_n)$ in a target language based on a training corpus
  - We still use LMs, just neural LMs for the most part

Statistical machine translation (SMT)

- Uses both a language model $P(y)$ and translation model $P(y|x)$ models the probability that y is a good translation for x
  - $$Pr(y|x) = \frac{Pr(y)Pr(x|y)}{Pr(x)}.$$
  - $$\hat{y} = \arg\max_y Pr(x|y)Pr(y)$$
- Language models (LMs) estimate the probability of an utterance $P(y) = P(w_1...w_n)$ in a target language based on a training corpus
  - We still use LMs, just neural LMs for the most part
  - Traditional approaches to estimating probability of a sentence were built on word frequency (available going back to the 40s, used for TEFOL)

Statistical machine translation (SMT)

- Uses both a language model $P(y)$ and translation model $P(y|x)$ models the probability that y is a good translation for x
  - $$Pr(y|x) = \frac{Pr(y)Pr(x|y)}{Pr(x)}.$$
  - $$\hat{y} = \arg\max_y Pr(x|y)Pr(y)$$
- Language models (LMs) estimate the probability of an utterance $P(y) = P(w_1...w_n)$ in a target language based on a training corpus
  - We still use LMs, just neural LMs for the most part
  - Traditional approaches to estimating probability of a sentence were built on word frequency (available going back to the 40s, used for TEFOL)
  - Then n-grams came along to better model linearity and dependencies

Statistical machine translation (SMT)

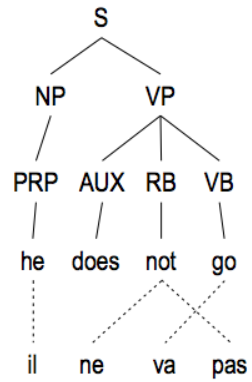
- Uses both a language model $P(y)$ and translation model $P(y|x)$ models the probability that y is a good translation for x
  - $$Pr(y|x) = \frac{Pr(y)Pr(x|y)}{Pr(x)}.$$
  - $$\hat{y} = \arg\max_y Pr(x|y)Pr(y)$$
- Language models (LMs) estimate the probability of an utterance $P(y) = P(w_1...w_n)$ in a target language based on a training corpus
  - We still use LMs, just neural LMs for the most part
  - Traditional approaches to estimating probability of a sentence were built on word frequency (available going back to the 40s, used for TEFOL)
  - Then n-grams came along to better model linearity and dependencies
  - Then lots of n-gram tricks (e.g. backoff and skipgrams) were invented to be more robust to sparsity and provide better estimates for the true probability of the sentence

SMT Cont.

- Translation models try to transform one sentence/phrase representation into another

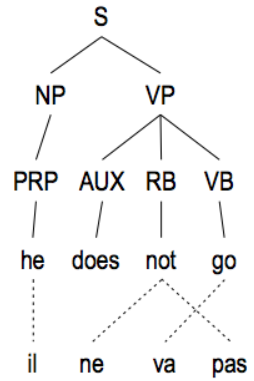
SMT Cont.

- Translation models try to transform one sentence/phrase representation into another
- Can be done on syntax trees:



SMT Cont.

- Translation models try to transform one sentence/phrase representation into another
- Can be done on syntax trees:

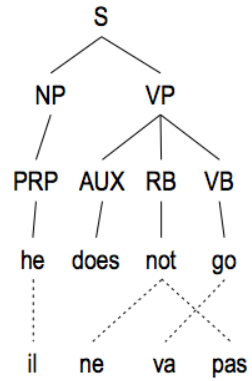


- Or done at the phrasal level (using a learned weighting on *parallelized* sentences called a "phrase table")

$$Pr(\phi(x,a,y)|x) = \frac{exp(w.\tilde{\phi}(x,a,y))}{\sum_{\tilde{\phi}\in\Phi_x} exp(w\cdot\tilde{\phi})},$$

SMT Cont.

- Translation models try to transform one sentence/phrase representation into another
- Can be done on syntax trees:



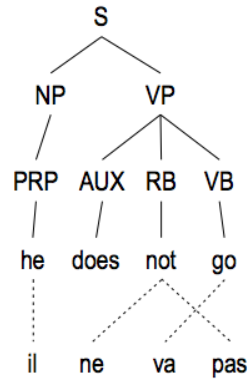
- Or done at the phrasal level (using a learned weighting on *parallelized* sentences called a "phrase table")

$$Pr(\phi(x,a,y)|x) = \frac{exp(w.\tilde{\phi}(x,a,y))}{\sum_{\tilde{\phi}\in\Phi_x} exp(w\cdot\tilde{\phi})},$$

- Phrase tables address the issue of reordering (i.e. "alignment") and non-1-to-1 mappings of words (e.g. ne... pas) between biphrases (the translation pair) seen in the tree above

SMT Cont.

- Translation models try to transform one sentence/phrase representation into another
- Can be done on syntax trees:



- Or done at the phrasal level (using a learned weighting on *parallelized* sentences called a "phrase table")

$$Pr(\phi(x,a,y)|x) = \frac{exp(w.\tilde{\phi}(x,a,y))}{\sum_{\tilde{\phi}\in\Phi_x} exp(w\cdot\tilde{\phi})},$$

- Phrase tables address the issue of reordering (i.e. "alignment") and non-1-to-1 mappings of words (e.g. ne... pas) between biphrases (the translation pair) seen in the tree above
- Weights are learned using SGD on a feature mapping to maximize MAP of the translation weights between the biphrases

Automatic evaluations of translation

- BLEU is most commonly used metric to automatically evaluate translation

Automatic evaluations of translation

- BLEU is most commonly used metric to automatically evaluate translation
- Takes clipped n-gram (clipping cannot exceed max n-gram count in any reference)

Candidate: the the the the the the the.
Reference 1: The cat is on the mat.
Reference 2: There is a cat on the mat.
Modified Unigram Precision = 2/7.[3]

Automatic evaluations of translation

- BLEU is most commonly used metric to automatically evaluate translation
- Takes clipped n-gram (clipping cannot exceed max n-gram count in any reference)

Candidate: the the the the the the the.
Reference 1: The cat is on the mat.
Reference 2: There is a cat on the mat.
Modified Unigram Precision = 2/7.[3]

- Sums over n-grams and over candidates

$$p_n = \frac{\sum_{C \in \{Candidates\}} \sum_{n\text{-}gram \in C} Count_{clip}(n\text{-}gram)}{\sum_{C' \in \{Candidates\}} \sum_{n\text{-}gram' \in C'} Count(n\text{-}gram')}.$$

Automatic evaluations of translation

- BLEU is most commonly used metric to automatically evaluate translation
- Takes clipped n-gram (clipping cannot exceed max n-gram count in any reference)

Candidate: the the the the the the the.
Reference 1: The cat is on the mat.
Reference 2: There is a cat on the mat.
Modified Unigram Precision = 2/7.[3]

- Sums over n-grams and over candidates

$$p_n = \frac{\sum_{C \in \{Candidates\}} \sum_{n\text{-}gram \in C} Count_{clip}(n\text{-}gram)}{\sum_{C' \in \{Candidates\}} \sum_{n\text{-}gram' \in C'} Count(n\text{-}gram')}.$$

- Enforces brevity penalty (BP) where r is best match (reference) length and c is candidate length (summed over all sentences in the document)

$$\mathrm{BP} = \begin{cases} 1 & \text{if } c > r \\ e^{(1-r/c)} & \text{if } c \leq r \end{cases}.$$

Automatic evaluations of translation

- BLEU is most commonly used metric to automatically evaluate translation
- Takes clipped n-gram (clipping cannot exceed max n-gram count in any reference)

Candidate: the the the the the the the.
Reference 1: The cat is on the mat.
Reference 2: There is a cat on the mat.
Modified Unigram Precision = 2/7.[3]

- Sums over n-grams and over candidates

$$p_n = \frac{\sum_{C \in \{Candidates\}} \sum_{n\text{-}gram \in C} Count_{clip}(n\text{-}gram)}{\sum_{C' \in \{Candidates\}} \sum_{n\text{-}gram' \in C'} Count(n\text{-}gram')}.$$

- Enforces brevity penalty (BP) where r is best match (reference) length and c is candidate length (summed over all sentences in the document)

$$\mathrm{BP} = \begin{cases} 1 & \text{if } c > r \\ e^{(1-r/c)} & \text{if } c \leq r \end{cases}.$$

$$\log \mathrm{BLEU} = \min(1 - \frac{r}{c}, 0) + \sum_{n=1}^{N} w_n \log p_n.$$

In our baseline, we use $N = 4$ and uniform weights $w_n = 1/N$.

- Then sums over ngrams

Text Classification

Naive Bayes Classifier

- Came about in the 60s with the rise of statistical data-modeling (rather than rule-based) methods

Naive Bayes Classifier

- Came about in the 60s with the rise of statistical data-modeling (rather than rule-based) methods
- Uses words as a whole (discrete units) and as they are (untransformed representations, though preprocessing is possible: e.g. stemmers/lemmatizers) as features to classify the text

Naive Bayes Classifier

- Came about in the 60s with the rise of statistical data-modeling (rather than rule-based) methods
- Uses words as a whole (discrete units) and as they are (untransformed representations, though preprocessing is possible: e.g. stemmers/lemmatizers) as features to classify the text
- Because it simply uses words (or n-grams), it's linear time to train (sounds nice...) by using a closed-form MLE to fit the classifier

Naive Bayes Classifier

- Came about in the 60s with the rise of statistical data-modeling (rather than rule-based) methods
- Uses words as a whole (discrete units) and as they are (untransformed representations, though preprocessing is possible: e.g. stemmers/lemmatizers) as features to classify the text
- Because it simply uses words (or n-grams), it's linear time to train (sounds nice...) by using a closed-form MLE to fit the classifier
- Reason it is called "naive" is that it assumes all features are independent in order to vastly simplify calculations, training time, complexity, etc. (definitely not true, but seems to not harm performance that much)

Naive Bayes Classifier

- Came about in the 60s with the rise of statistical data-modeling (rather than rule-based) methods
- Uses words as a whole (discrete units) and as they are (untransformed representations, though preprocessing is possible: e.g. stemmers/lemmatizers) as features to classify the text
- Because it simply uses words (or n-grams), it's linear time to train (sounds nice...) by using a closed-form MLE to fit the classifier
- Reason it is called "naive" is that it assumes all features are independent in order to vastly simplify calculations, training time, complexity, etc. (definitely not true, but seems to not harm performance that much)
- Because of the model simplicity, it needs way less training data

Naive Bayes Classifier

- Came about in the 60s with the rise of statistical data-modeling (rather than rule-based) methods
- Uses words as a whole (discrete units) and as they are (untransformed representations, though preprocessing is possible: e.g. stemmers/lemmatizers) as features to classify the text
- Because it simply uses words (or n-grams), it's linear time to train (sounds nice...) by using a closed-form MLE to fit the classifier
- Reason it is called "naive" is that it assumes all features are independent in order to vastly simplify calculations, training time, complexity, etc. (definitely not true, but seems to not harm performance that much)
- Because of the model simplicity, it needs way less training data
- Easy to train, you just need to count words and count labeled documents, maybe use stop words (close-class words etc.)

Naive Bayes Math

- Start with base equation

$$p(C_k \mid \mathbf{x}) = \frac{p(C_k)\, p(\mathbf{x} \mid C_k)}{p(\mathbf{x})}$$

Naive Bayes Math

- Start with base equation

$$p(C_k \mid \mathbf{x}) = \frac{p(C_k)\, p(\mathbf{x} \mid C_k)}{p(\mathbf{x})}$$

- Ditch denomenator and assume independence

$$\begin{aligned} p(C_k \mid x_1, \ldots, x_n) &\propto p(C_k, x_1, \ldots, x_n) \\ &= p(C_k)\, p(x_1 \mid C_k)\, p(x_2 \mid C_k)\, p(x_3 \mid C_k) \cdots \\ &= p(C_k) \prod_{i=1}^{n} p(x_i \mid C_k)\,, \end{aligned}$$

Naive Bayes Classification

- Choose the class that maximizes your likelihood

$$\hat{y} = \underset{k \in \{1,\ldots,K\}}{\operatorname{argmax}} \; p(C_k) \prod_{i=1}^{n} p(x_i \mid C_k).$$

Naive Bayes Classification

- Choose the class that maximizes your likelihood

$$\hat{y} = \underset{k \in \{1,\ldots,K\}}{\operatorname{argmax}} \; p(C_k) \prod_{i=1}^{n} p(x_i \mid C_k).$$

- In the multinomial case, this is a linear classifier

$$\begin{aligned} \log p(C_k \mid \mathbf{x}) &\propto \log \left(p(C_k) \prod_{i=1}^{n} {p_{ki}}^{x_i} \right) \\ &= \log p(C_k) + \sum_{i=1}^{n} x_i \cdot \log p_{ki} \\ &= b + \mathbf{w}_k^\top \mathbf{x} \end{aligned}$$

where $b = \log p(C_k)$ and $w_{ki} = \log p_{ki}$.

Grammar Parsing, etc.

Chart Parsing

- Attempts to overcome the fact that parse trees are mutually exclusive and often cannot share information between them (they have completely different structures)

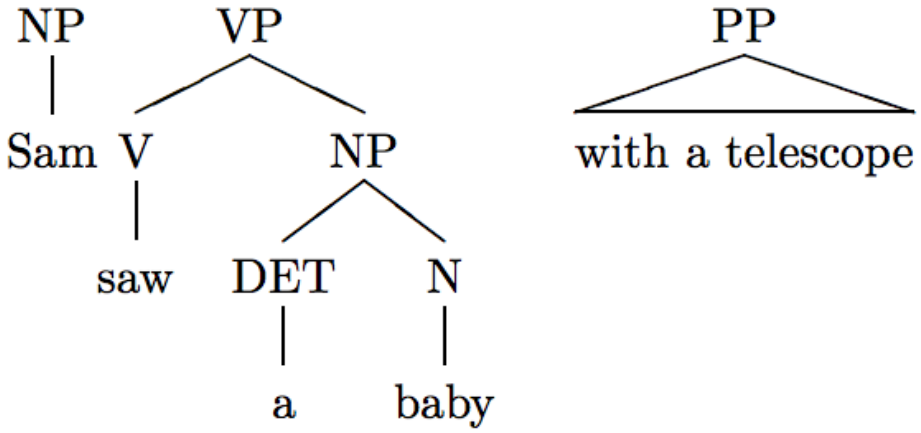
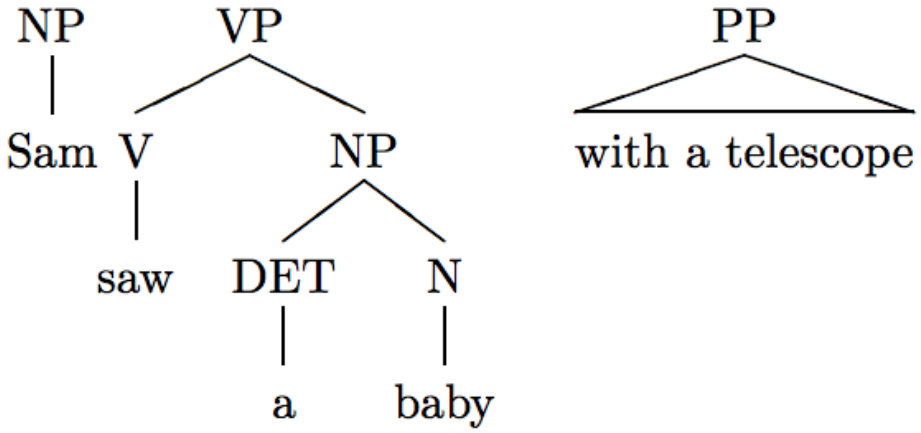


Chart Parsing

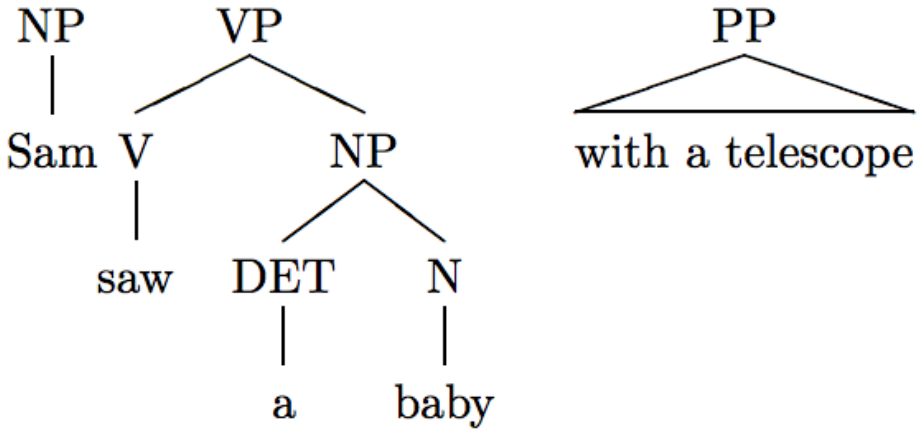
- Attempts to overcome the fact that parse trees are mutually exclusive and often cannot share information between them (they have completely different structures)



- Additionally, before they achieve closure, they're not really possible or meaningful to store as individual constituents

Chart Parsing

- Attempts to overcome the fact that parse trees are mutually exclusive and often cannot share information between them (they have completely different structures)



- Additionally, before they achieve closure, they're not really possible or meaningful to store as individual constituents
- Uses dynamic programming to build solutions that both can store different intermediary steps (taking care of pre-closure representations) and represent different paths to solutions (taking care of separate storage of ambiguity)

Quick Dynamic Programming Review

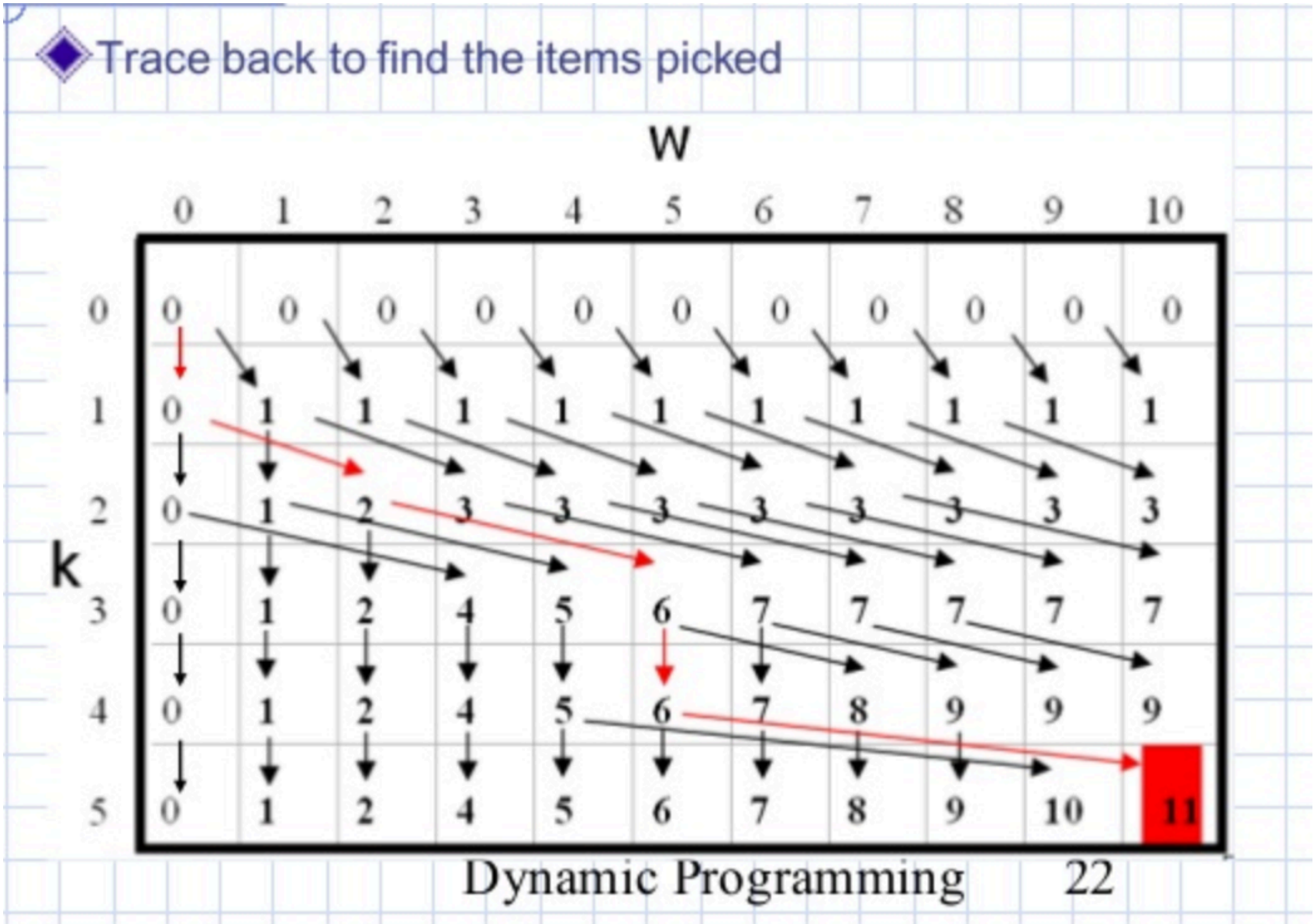


Figure 5: Dynamic Programming

Chart Parsing Cont.

- Represents sentences as a DAG with word and POS arcs

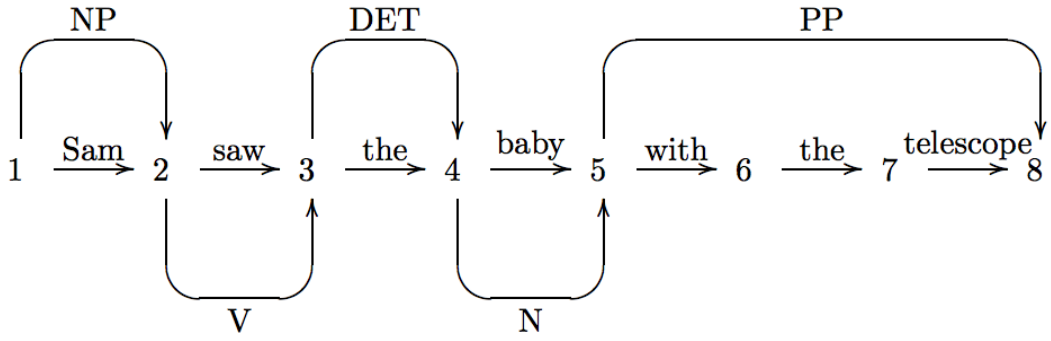
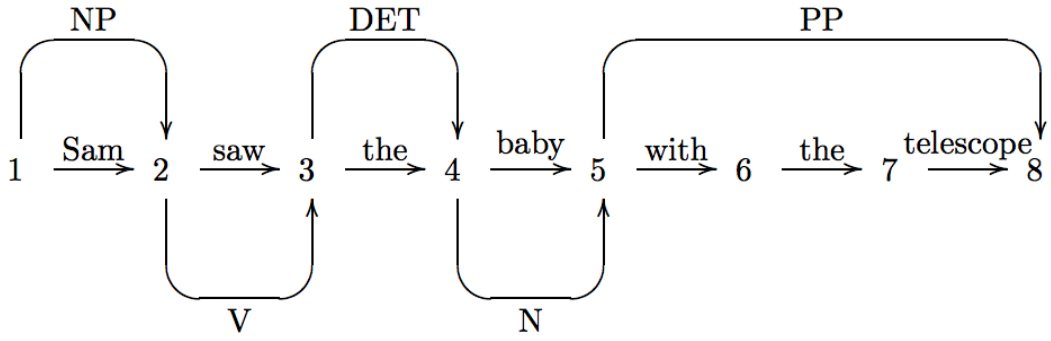
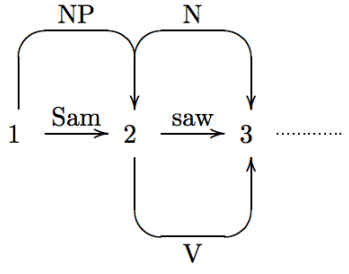


Chart Parsing Cont.

- Represents sentences as a DAG with word and POS arcs

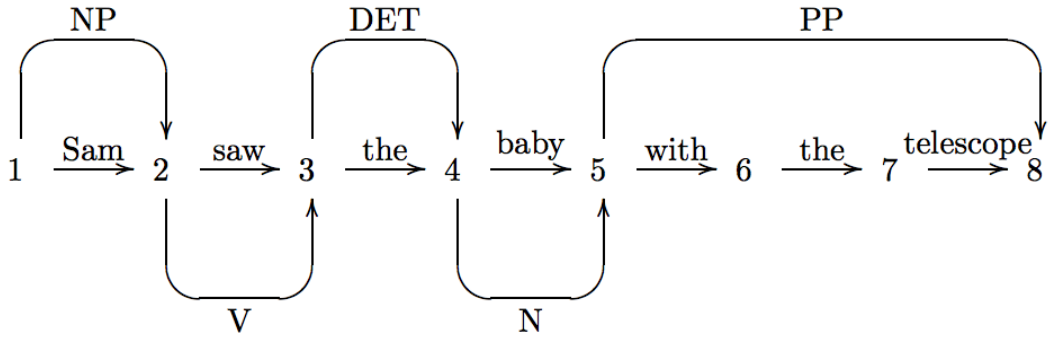


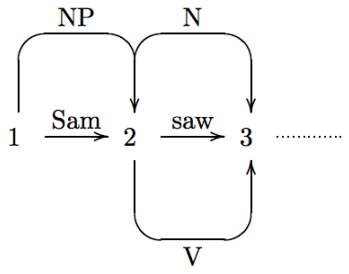


- Allows incomplete parses in the chart

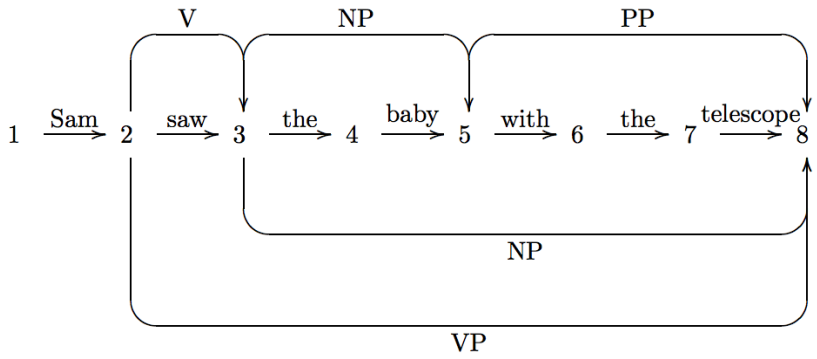
Chart Parsing Cont.

- Represents sentences as a DAG with word and POS arcs





- Allows incomplete parses in the chart
- Allows multiple correct parses to be stored in the graph



Word Sense Disambiguation

WordNet

- Built in 1995 by George Miller and a bunch of other contributors by mining many corpora of English (usually built on word counts of literary published texts, e.g. Kucera and Francis)

WordNet

- Built in 1995 by George Miller and a bunch of other contributors by mining many corpora of English (usually built on word counts of literary published texts, e.g. Kucera and Francis)
- It's a lexical ontology database that's inspired by psycholinguistic theories about how lexical memory is structured (theory network, e.g. see Steven Pinker's Royal Institute YouTube talk), that gives rise to things like "word association" games

WordNet

- Built in 1995 by George Miller and a bunch of other contributors by mining many corpora of English (usually built on word counts of literary published texts, e.g. Kucera and Francis)
- It's a lexical ontology database that's inspired by psycholinguistic theories about how lexical memory is structured (theory network, e.g. see Steven Pinker's Royal Institute YouTube talk), that gives rise to things like "word association" games
- Contains entry for all the senses of a given word and hyperlinks to synonyms and antonyms for each sense

WordNet

- Built in 1995 by George Miller and a bunch of other contributors by mining many corpora of English (usually built on word counts of literary published texts, e.g. Kucera and Francis)
- It's a lexical ontology database that's inspired by psycholinguistic theories about how lexical memory is structured (theory network, e.g. see Steven Pinker's Royal Institute YouTube talk), that gives rise to things like "word association" games
- Contains entry for all the senses of a given word and hyperlinks to synonyms and antonyms for each sense
- Also contains for nouns hypernyms and hyponyms (superclass/subclass), meronyms and holonyms (is part of/contains); for verbs hypernym, troponym (specific way of doing something), entailment (necessary precondition)

WordNet UI

Word to search for: plant Search WordNet

Display Options: (Select option to change) Change

Key: "S:" = Show Synset (semantic) relations, "W:" = Show Word (lexical) relations

Display options for sense: (gloss) "an example sentence"

Noun

- S: (n) **plant**, works, industrial plant (buildings for carrying on industrial labor) *"they built a large plant to manufacture automobiles"*
- S: (n) **plant**, flora, plant life ((botany) a living organism lacking the power of locomotion)
- S: (n) **plant** (an actor situated in the audience whose acting is rehearsed but seems spontaneous to the audience)
- S: (n) **plant** (something planted secretly for discovery by another) *"the police used a plant to trick the thieves"; "he claimed that the evidence against him was a plant"*

Verb

- S: (v) **plant**, set (put or set (seeds, seedlings, or plants) into the ground) *"Let's plant flowers in the garden"*
- S: (v) implant, engraft, embed, imbed, **plant** (fix or set securely or deeply) *"He planted a knee in the back of his opponent"; "The dentist implanted a tooth in the gum"*
- S: (v) establish, found, **plant**, constitute, institute (set up or lay the groundwork for) *"establish a new department"*
- S: (v) **plant** (place into a river) *"plant fish"*
- S: (v) **plant** (place something or someone in a certain position in order to secretly observe or deceive) *"Plant a spy in Moscow"; "plant bugs in the dissident's apartment"*

WordNet Cont.

- In this way, it creates an undirected unweighted (or bivalent weighted if you include antonyms) graph network that represents word sense relationships

WordNet Cont.

- In this way, it creates an undirected unweighted (or bivalent weighted if you include antonyms) graph network that represents word sense relationships
- Given this database, you can do the numerous analyses that are possible on graphs to learn relations in the network

WordNet Cont.

- In this way, it creates an undirected unweighted (or bivalent weighted if you include antonyms) graph network that represents word sense relationships
- Given this database, you can do the numerous analyses that are possible on graphs to learn relations in the network
- A simple approach might be to substitute in all the possible synonyms of each sense and see how many of those are attested in the corpus

Neural Approaches:

The problem

- Seeks to solve the vanishing/exploding gradient problem (don't they all...): i.e. given N layers to a network, the gradient is raised to the N by the time it gets back to the first layer

The problem

- Seeks to solve the vanishing/exploding gradient problem (don't they all...): i.e. given N layers to a network, the gradient is raised to the N by the time it gets back to the first layer
- This is particularly a problem for recurrent neural networks (RNNs) because back propogation through time (BPTT) means that the networks are as deep as their sequence is long. So given a sentence of N words, there are essentially N layers to the network (and that's assuming each time state only has 1 layer)

The problem

- Seeks to solve the vanishing/exploding gradient problem (don't they all...): i.e. given N layers to a network, the gradient is raised to the N by the time it gets back to the first layer
- This is particularly a problem for recurrent neural networks (RNNs) because back propogation through time (BPTT) means that the networks are as deep as their sequence is long. So given a sentence of N words, there are essentially N layers to the network (and that's assuming each time state only has 1 layer)
- In 1997 Hochreiter and Schmidhuber come up with a solution

LSTM V1

- Introduces the constant error carousel (CEC)

LSTM V1

- Introduces the constant error carousel (CEC)
  - Selectively allows error to flow back through the network for an arbitrarily long amount of time (kind of resnet-y)

LSTM V1

- Introduces the constant error carousel (CEC)
  - Selectively allows error to flow back through the network for an arbitrarily long amount of time (kind of resnet-y)
  - Determined by gating functions that are concurrently trained/learned

LSTM V1

- Introduces the constant error carousel (CEC)
  - Selectively allows error to flow back through the network for an arbitrarily long amount of time (kind of resnet-y)
  - Determined by gating functions that are concurrently trained/learned
  - Termed "gates" because they determine how much error flows back and how much memory flows forward, like the gates in water lochs

LSTM V1

- Introduces the constant error carousel (CEC)
  - Selectively allows error to flow back through the network for an arbitrarily long amount of time (kind of resnet-y)
  - Determined by gating functions that are concurrently trained/learned
  - Termed "gates" because they determine how much error flows back and how much memory flows forward, like the gates in water lochs
- How to solve the vanishing/exploding gradient?

LSTM V1

- Introduces the constant error carousel (CEC)
  - Selectively allows error to flow back through the network for an arbitrarily long amount of time (kind of resnet-y)
  - Determined by gating functions that are concurrently trained/learned
  - Termed "gates" because they determine how much error flows back and how much memory flows forward, like the gates in water lochs
- How to solve the vanishing/exploding gradient?
  - Don't backpropogate (at least not all the way...)

LSTM V1

- Introduces the constant error carousel (CEC)
  - Selectively allows error to flow back through the network for an arbitrarily long amount of time (kind of resnet-y)
  - Determined by gating functions that are concurrently trained/learned
  - Termed "gates" because they determine how much error flows back and how much memory flows forward, like the gates in water lochs
- How to solve the vanishing/exploding gradient?
  - Don't backpropogate (at least not all the way...)
  - Backprop only for the current time-step so it's a reasonable decay of gradient information, let the CEC take care of error propogation across timesteps

LSTM V1

- Introduces the constant error carousel (CEC)
  - Selectively allows error to flow back through the network for an arbitrarily long amount of time (kind of resnet-y)
  - Determined by gating functions that are concurrently trained/learned
  - Termed "gates" because they determine how much error flows back and how much memory flows forward, like the gates in water lochs
- How to solve the vanishing/exploding gradient?
  - Don't backpropogate (at least not all the way...)
  - Backprop only for the current time-step so it's a reasonable decay of gradient information, let the CEC take care of error propogation across timesteps
- Form "memory cell" units that have internal topologies but all receive the same input from the input gate and whose combined outputs are fed to the output gate

LSTM Gates Semantics

- Forward Pass

LSTM Gates Semantics

- Forward Pass
  - Input gate protects memory (and in back-pass error) from irrelevant/undesirable inputs: how much incoming information (input) should I allow into my hidden layers (memory)?

LSTM Gates Semantics

- Forward Pass
  - Input gate protects memory (and in back-pass error) from irrelevant/undesirable inputs: how much incoming information (input) should I allow into my hidden layers (memory)?
  - Output gate protects downstream units from irrelevant memory contents (i.e. hidden states): how much hidden information (memory) should I pass downstream to influence future cells?

LSTM Gates Semantics

- Forward Pass
  - Input gate protects memory (and in back-pass error) from irrelevant/undesirable inputs: how much incoming information (input) should I allow into my hidden layers (memory)?
  - Output gate protects downstream units from irrelevant memory contents (i.e. hidden states): how much hidden information (memory) should I pass downstream to influence future cells?
- Backward Pass

LSTM Gates Semantics

- Forward Pass
  - Input gate protects memory (and in back-pass error) from irrelevant/undesirable inputs: how much incoming information (input) should I allow into my hidden layers (memory)?
  - Output gate protects downstream units from irrelevant memory contents (i.e. hidden states): how much hidden information (memory) should I pass downstream to influence future cells?
- Backward Pass
  - Output gate decides how much error to keep in the CEC to prevent vanishing/exploding error

LSTM Gates Semantics

- Forward Pass
  - Input gate protects memory (and in back-pass error) from irrelevant/undesirable inputs: how much incoming information (input) should I allow into my hidden layers (memory)?
  - Output gate protects downstream units from irrelevant memory contents (i.e. hidden states): how much hidden information (memory) should I pass downstream to influence future cells?
- Backward Pass
  - Output gate decides how much error to keep in the CEC to prevent vanishing/exploding error
  - Input gate decided how much error to throw away before passing it on to the previous time step/memory cell

LSTM Gates Syntax

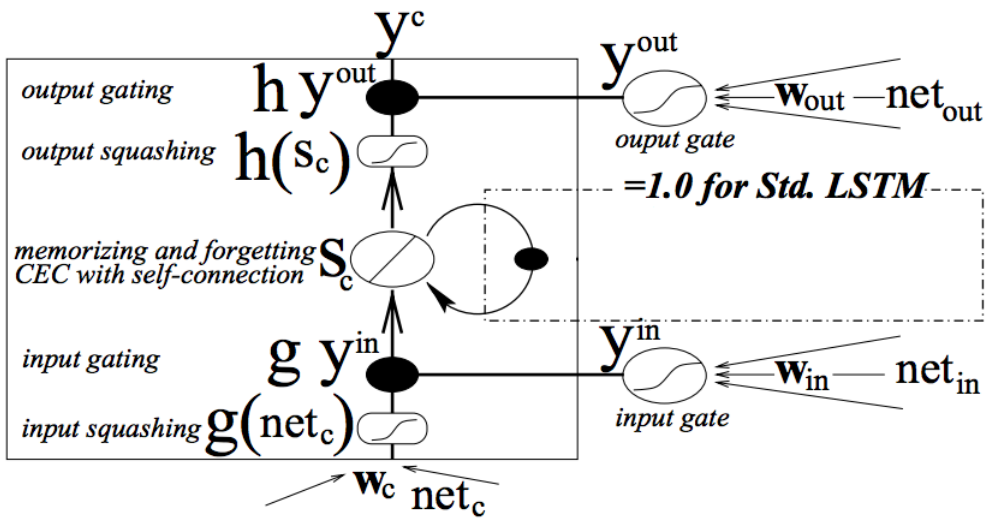


Figure 1: The standard LSTM cell has a linear unit with a recurrent self-connection with weight 1.0 (CEC). Input and output gates regulate read and write access to the cell whose state is denoted s_c. The function g squashes the cell's input; h squashes the cell's output. See the text for details.

$$
\begin{aligned}
i_t &= \sigma_g(W_i x_t + U_i h_{t-1} + b_i) \\
o_t &= \sigma_g(W_o x_t + U_o h_{t-1} + b_o) \\
c_t &= \quad c_{t-1} + i_t \circ \sigma_c(W_c x_t + U_c h_{t-1} + b_c) \\
h_t &= o_t \circ \sigma_h(c_t)
\end{aligned}
$$

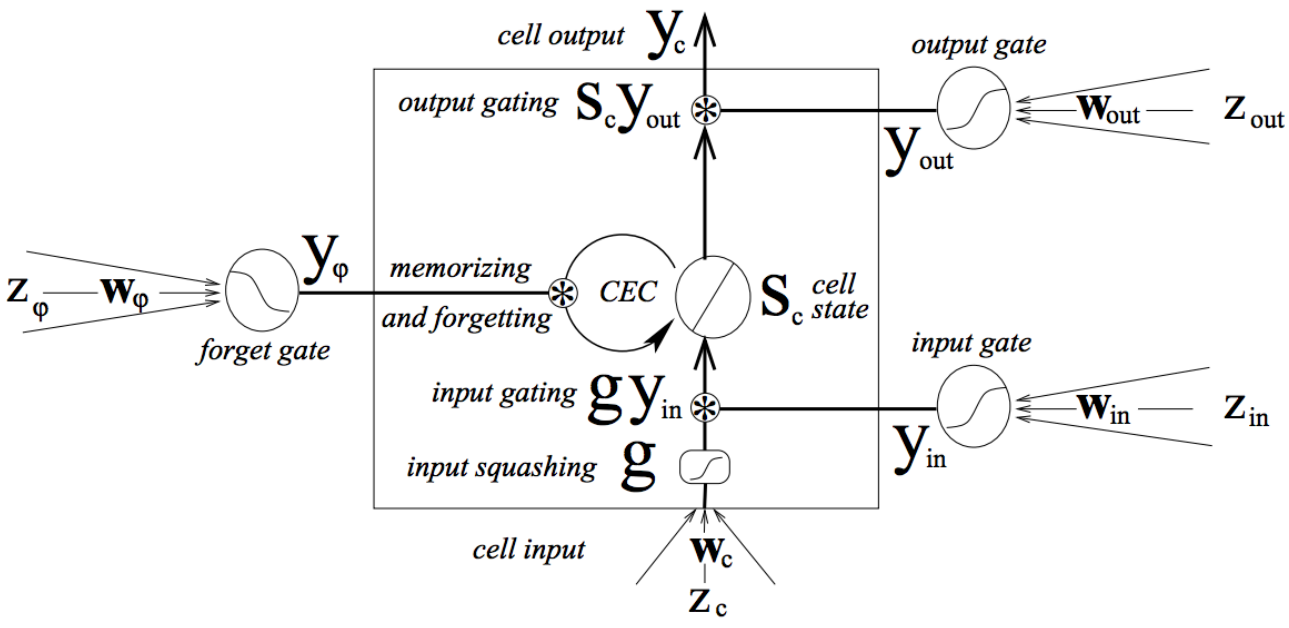
LSTM V2: Forgetting

- Rather than letting the error backpropogate unobstructed through the network, let the network decide what to let through the CEC

LSTM V2: Forgetting

- Rather than letting the error backpropogate unobstructed through the network, let the network decide what to let through the CEC
  - Add in forget gate that controls what is let through from the previous cell memory/hidden state

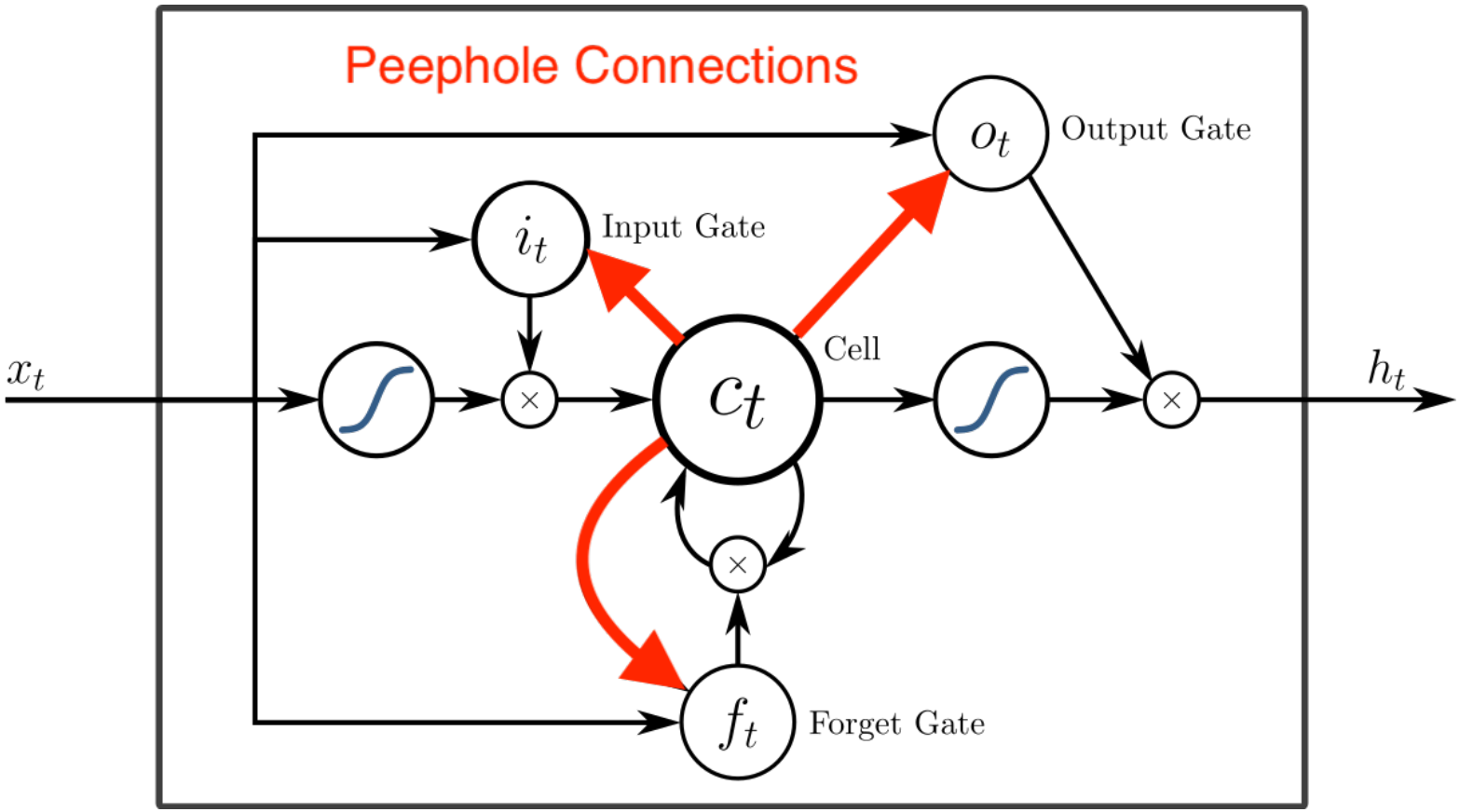
Forgetting LSTM



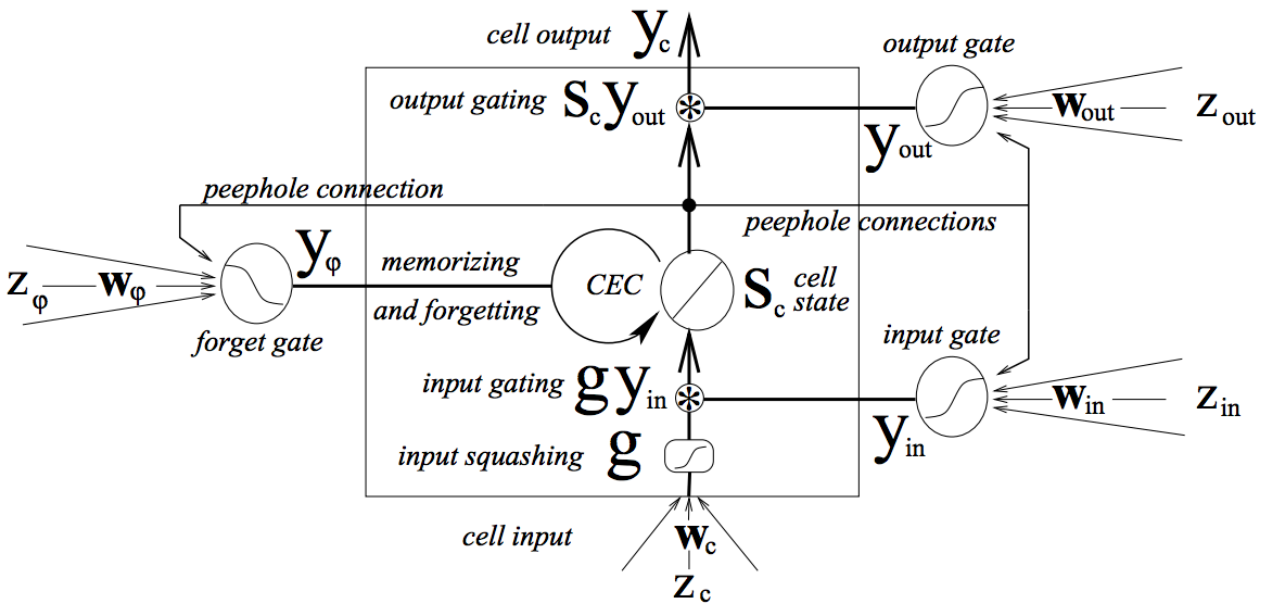
$$f_t = \sigma_g(W_f x_t + U_f h_{t-1} + b_f)$$
$$i_t = \sigma_g(W_i x_t + U_i h_{t-1} + b_i)$$
$$o_t = \sigma_g(W_o x_t + U_o h_{t-1} + b_o)$$
$$c_t = f_t \circ c_{t-1} + i_t \circ \sigma_c(W_c x_t + U_c h_{t-1} + b_c)$$
$$h_t = o_t \circ \sigma_h(c_t)$$

LSTM V3: Peephole

- Let the previous state of the memory cell directly influence what will be forgotten, let in, and let out



LSTM V3: Peephole Cont.



$$f_t = \sigma_g(W_f x_t + U_f c_{t-1} + b_f)$$
$$i_t = \sigma_g(W_i x_t + U_i c_{t-1} + b_i)$$
$$o_t = \sigma_g(W_o x_t + U_o c_{t-1} + b_o)$$
$$c_t = f_t \circ c_{t-1} + i_t \circ \sigma_c(W_c x_t + b_c)$$
$$h_t = \sigma_h(o_t \circ c_t)$$

LSTM: Marrying semantics and syntax

$$f_{t(m\times1)} = \sigma(W_{xf(m\times n)}x_{t(n\times1)} + W_{hf(m\times m)}h_{t-1(m\times1)} + b_{f(m\times1)})$$

$$c_{t(m\times1)} = i_{t(m\times1)} \circ .* u_{t(m\times1)} + f_{t(m\times1)} .* c_{t-1(m\times1)}$$

.* = pointwise multiplication

$$o_{t(m\times1)} = \sigma(W_{xo(m\times n)}x_{t(n\times1)} + W_{ho(m\times m)}h_{t-1(m\times1)} + b_{o(m\times1)})$$

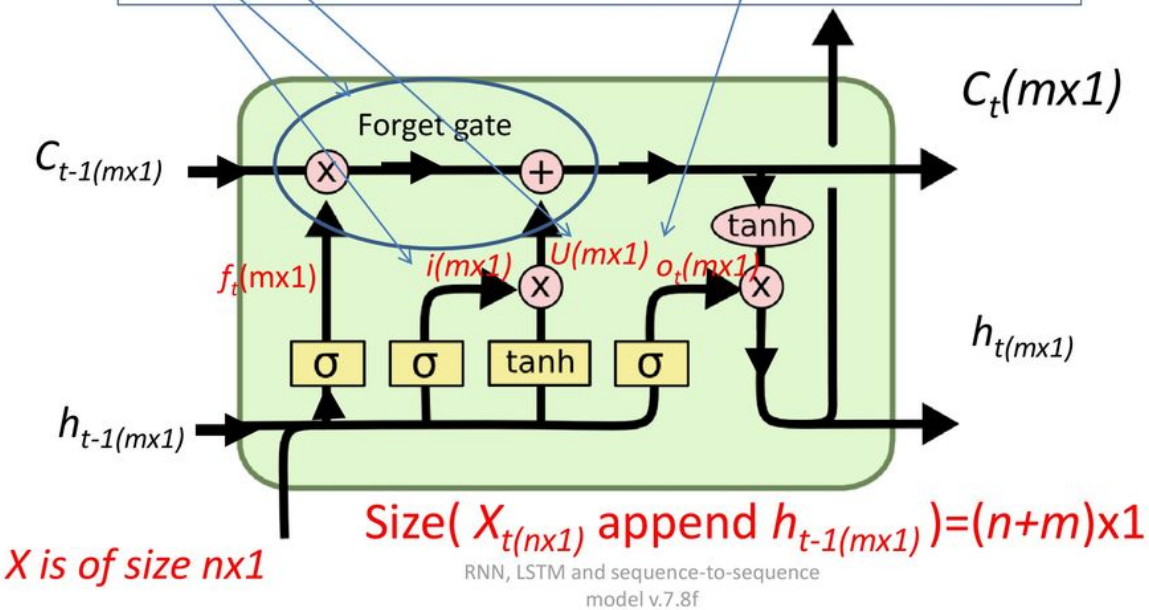
- $$u_{t(m\times1)} = \tanh(W_{xu(m\times n)}x_{t(n\times1)} + W_{hu(m\times m)}h_{t-1(m\times1)} + b_{u(m\times1)})$$

$$i_{t(m\times1)} = \sigma(W_{xi(m\times n)}x_{t(n\times1)} + W_{hi(m\times m)}h_{t-1(m\times1)} + b_{i(m\times1)})$$

X is of size nx1

h is of size mx1

http://kvitajakub.github.io/2016/04/14/rnn-diagrams/



LSTM V4: Coupled forget and input gates

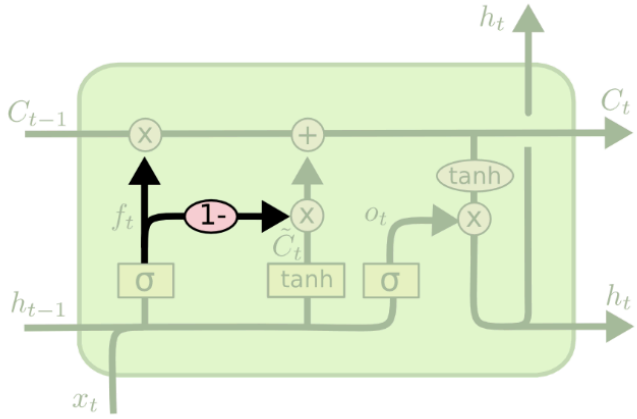
- Rather than having separate input and forgetting gates, why not make them complements of each other

LSTM V4: Coupled forget and input gates

- Rather than having separate input and forgetting gates, why not make them complements of each other
  - Get rid of input gate parameters and determine wholly from forget gate

LSTM V4: Coupled forget and input gates

- Rather than having separate input and forgetting gates, why not make them complements of each other
  - Get rid of input gate parameters and determine wholly from forget gate
  - Gives a tug-of-war relationship to information preservation vs. onboarding



$$C_t = f_t * C_{t-1} + (1 - \boldsymbol{f_t}) * \tilde{C}_t$$

Simplified LSTM: GRU

- Simplify, simplify, simplify (at the cost of power)

Simplified LSTM: GRU

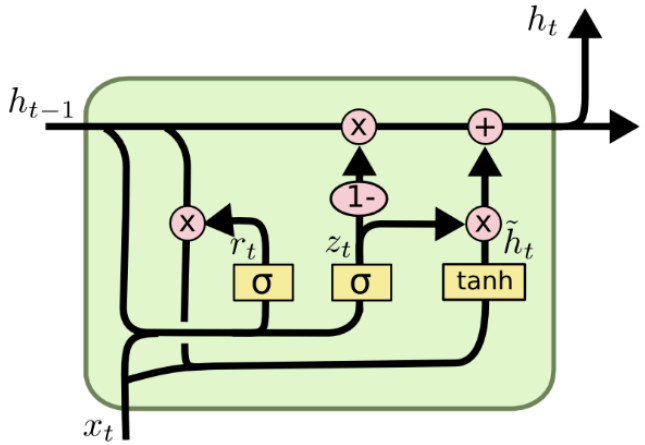
- Simplify, simplify, simplify (at the cost of power)
  - Combine forget and input gates (why stop at coupling them?) into "update gate" z_t

Simplified LSTM: GRU

- Simplify, simplify, simplify (at the cost of power)
  - Combine forget and input gates (why stop at coupling them?) into "update gate" z_t
  - Do away with separate hidden state and cell memory and have just hidden state

Simplified LSTM: GRU

- Simplify, simplify, simplify (at the cost of power)
  - Combine forget and input gates (why stop at coupling them?) into "update gate" z_t
  - Do away with separate hidden state and cell memory and have just hidden state
  - Introduce a reset vector to allow "forgetting"



$$z_t = \sigma\left(W_z \cdot [h_{t-1}, x_t]\right)$$

$$r_t = \sigma\left(W_r \cdot [h_{t-1}, x_t]\right)$$

$$\tilde{h}_t = \tanh\left(W \cdot [r_t * h_{t-1}, x_t]\right)$$

$$h_t = (1 - z_t) * h_{t-1} + z_t * \tilde{h}_t$$