

CS 546 – Advanced Topics in NLP

Dilek Hakkani-Tür



UNIVERSITY OF
ILLINOIS
URBANA-CHAMPAIGN



Siebel School of
Computing
and Data Science



Topics for Today

Model Architectures and Contextual Embeddings

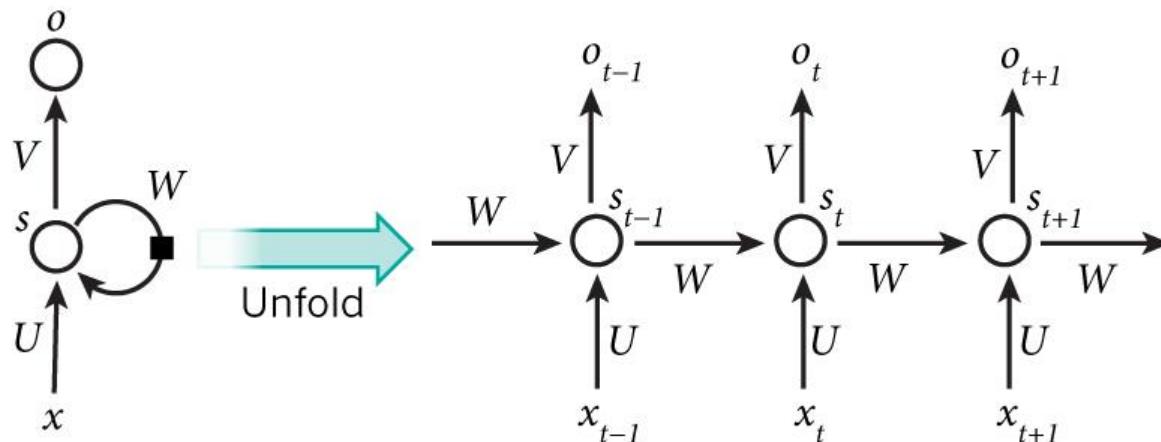
- Long-Short Term Memory (LSTM)
- Gated Recurrent Units (GRU)
- Example Sequence Classification Tasks
- Elmo and Contextual Embeddings

Recurrent Neural Networks (RNNs)

Slide from previous lecture

$$s_t = \sigma(W s_{t-1} + U x_t) \quad \sigma(\cdot): \text{tanh, ReLU}$$

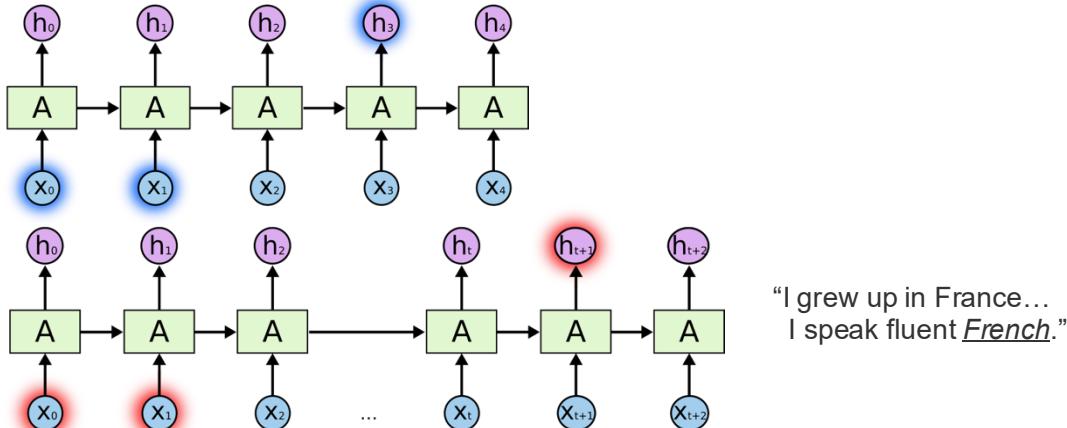
$$o_t = \text{softmax}(V s_t)$$



Gating Mechanisms: Solution to Vanishing Gradients

- RNN models temporal sequence information
 - can handle “long-term dependencies” *in theory*

Slide from previous lecture



Issue: RNN cannot handle such “long-term dependencies” in practice due to vanishing gradient
→ apply the gating mechanism to directly encode the long-distance information

Long Distance Dependencies



- What comes after the following context window?
“purred” or “barked”?

“.... the cat”

“purred”

“The dog that was chasing the cat”

“barked”

Irrelevant Segments



- What are the intent categories of the following utterances?
- I would like to set an alarm → Set_alarm
- I would like to send an email → Send_email
- I need to book a table → Reserve_restaurant
- I need to set an alarm → Set_alarm

Discontinuous Segments

- What is the topic of the following conversation?

A: Have you had a chance to finish homework 1?

B: I am almost done. **I heard they found a new case of coronavirus in the bay area today.**

A: **Yeah, I saw that too. We need to be careful to not spread it further.**

B: **Yes, absolutely.** I tried using Fasttext embeddings they helped a bit.

A: Oh, I am using Glove embeddings.

B: Yeah, I tried them as well.

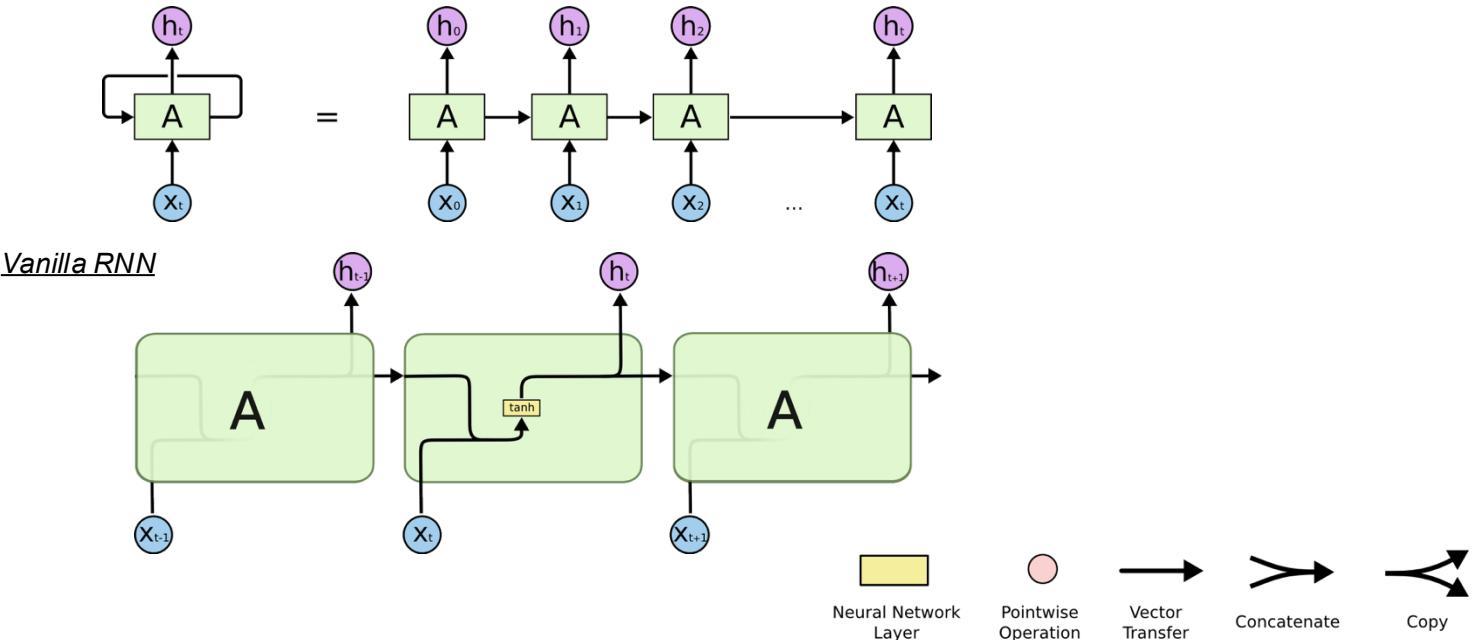
Gating the Hidden State



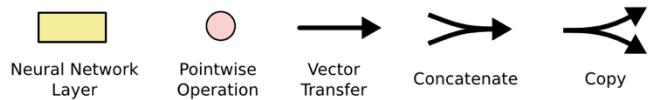
- Dedicated mechanisms, for example, when:
 - a hidden state should be updated,
 - it should be reset, and
 - information in hidden state should be used
- LSTMs: Gated memory cell and input, output, forget gates
- GRUs: update and reset gates.

Long Short-Term Memory (LSTM)

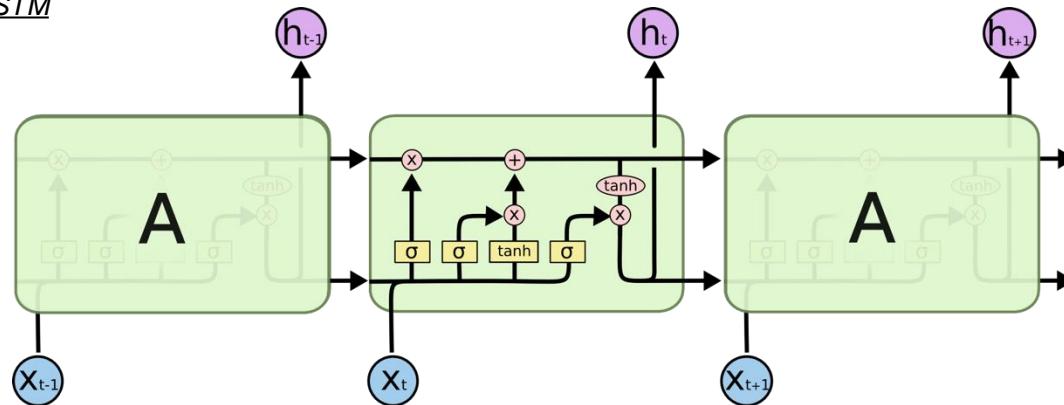
- LSTMs are explicitly designed to enable the long-term dependencies



Long Short-Term Memory (LSTM, cont.)

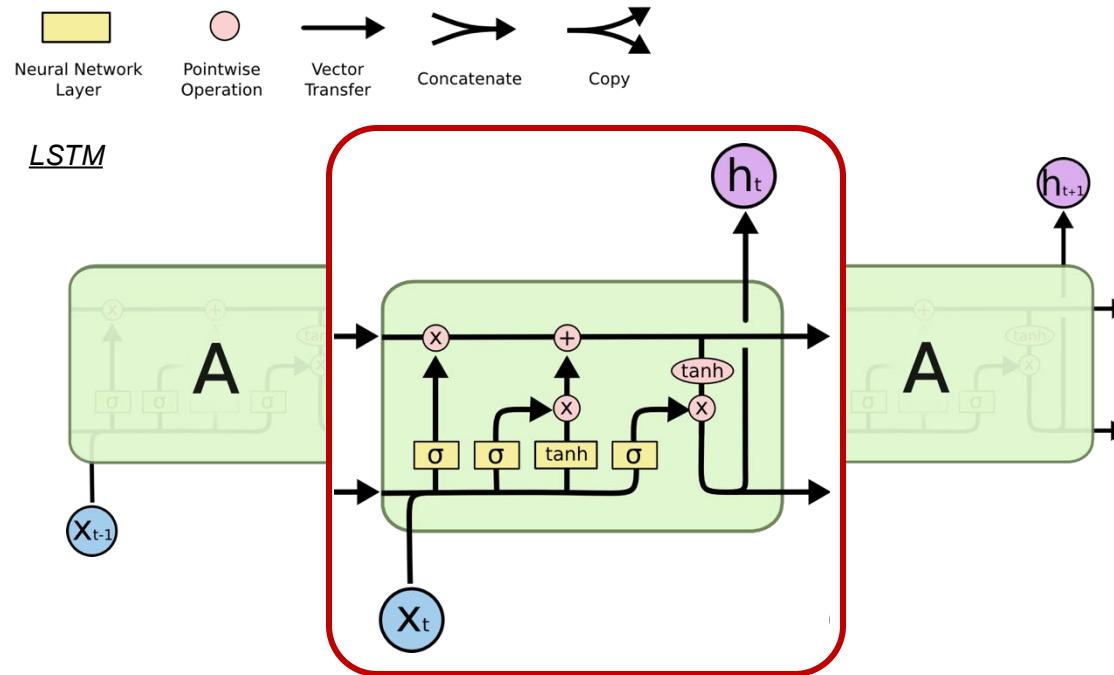


LSTM



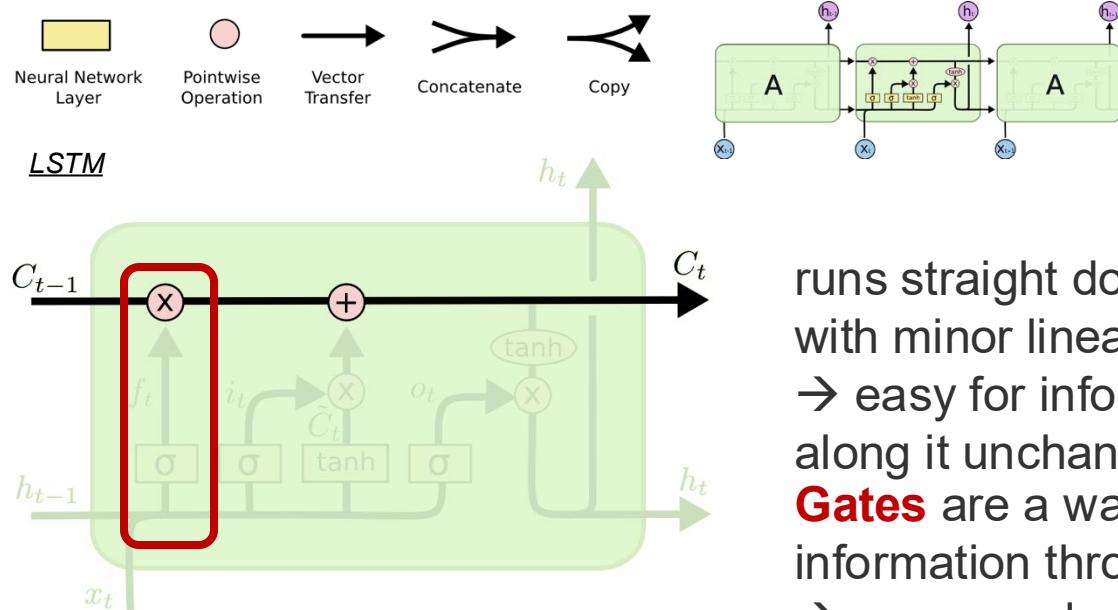
Hochreiter and Schmidhuber, "Long short-term memory," in Neural Computation, 1997. [[link](#)]

Long Short-Term Memory (LSTM, cont.)



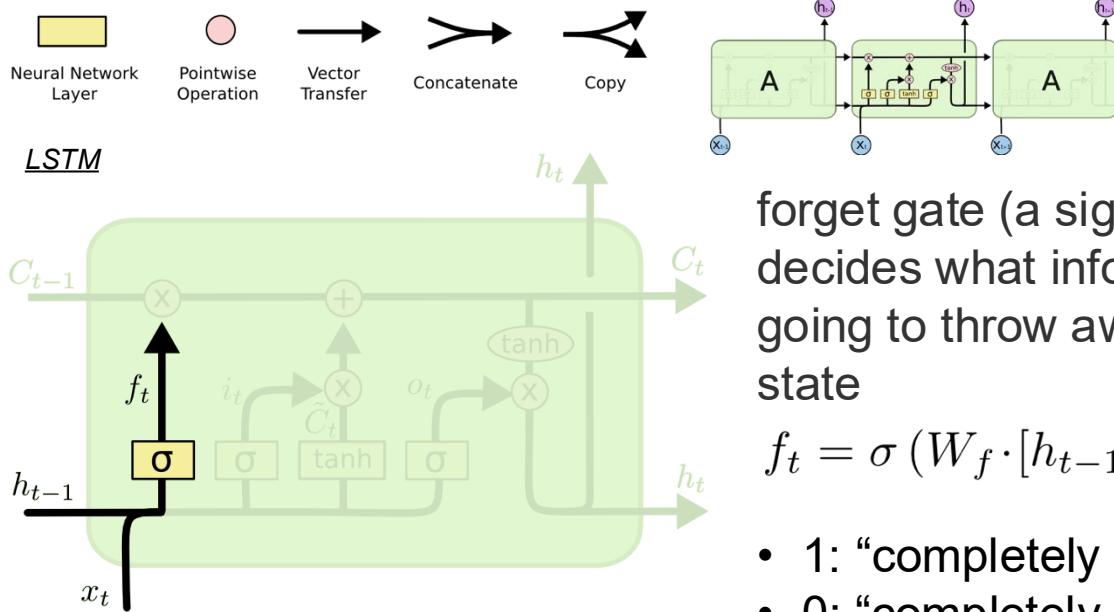
Hochreiter and Schmidhuber, "Long short-term memory," in Neural Computation, 1997. [[link](#)]

Long Short-Term Memory (LSTM, cont.)



runs straight down the chain
with minor linear interactions
→ easy for information to flow
along it unchanged
Gates are a way to optionally let
information through
→ composed of a sigmoid and a
pointwise multiplication operation

Long Short-Term Memory (LSTM, cont.)



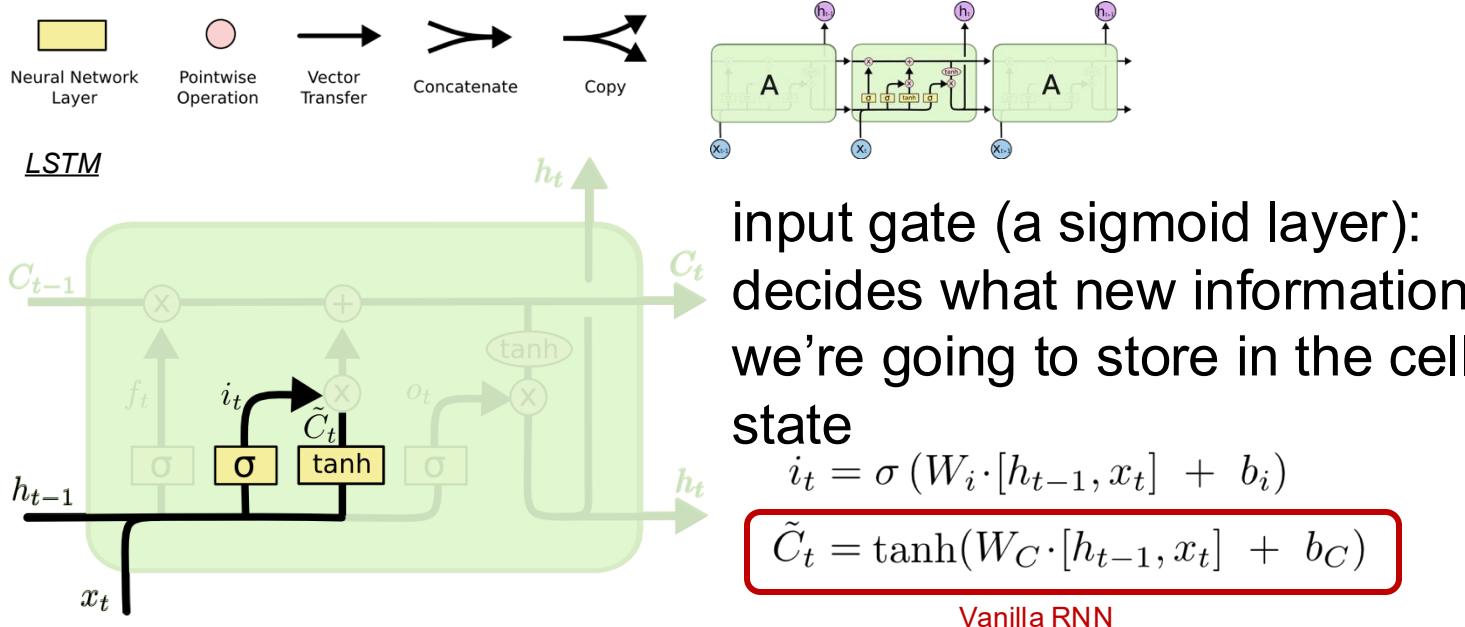
forget gate (a sigmoid layer):
decides what information we're
going to throw away from the cell
state

$$f_t = \sigma (W_f \cdot [h_{t-1}, x_t] + b_f)$$

- 1: “completely keep this”
- 0: “completely get rid of this”

Example: The cell state might include the gender of the present subject, so that the correct pronouns can be used. When seeing a new subject, we want to forget the old subject's gender.

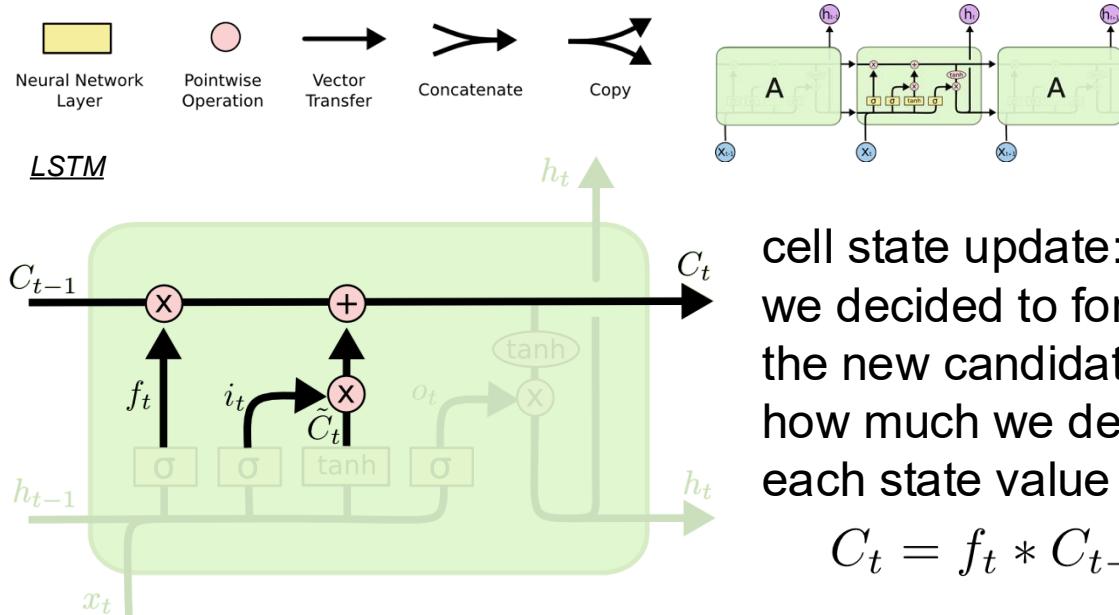
Long Short-Term Memory (LSTM, cont.)



Example: We want to add the new subject's gender to the cell state for replacing the old one.

Hochreiter and Schmidhuber, "Long short-term memory," in Neural Computation, 1997. [[link](#)]

Long Short-Term Memory (LSTM, cont.)



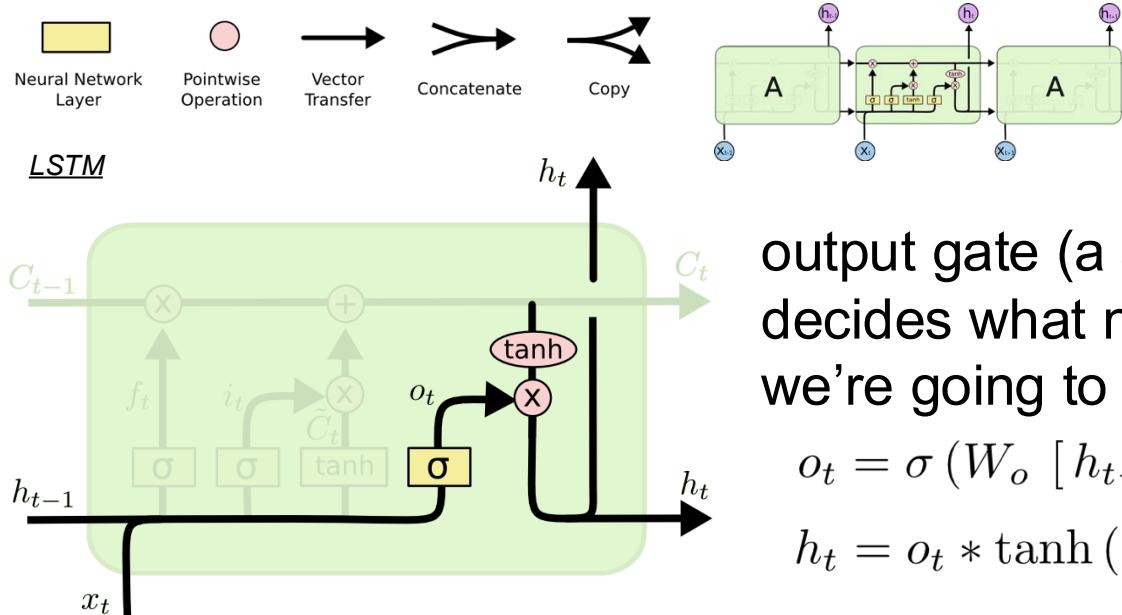
cell state update: forgets the things we decided to forget earlier and adds the new candidate values, scaled by how much we decided to update each state value

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

Example: where we actually drop the information about the old subject's gender and add the new information

- f_t : decides what to forget
- i_t : decides what to update

Long Short-Term Memory (LSTM, cont.)



output gate (a sigmoid layer):
decides what new information
we're going to output

$$o_t = \sigma (W_o [h_{t-1}, x_t] + b_o)$$

$$h_t = o_t * \tanh (C_t)$$

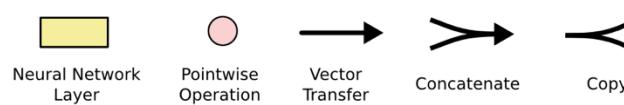
Example: It might output whether the subject is singular or plural given the context

LSTM Variants

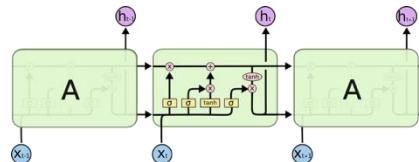
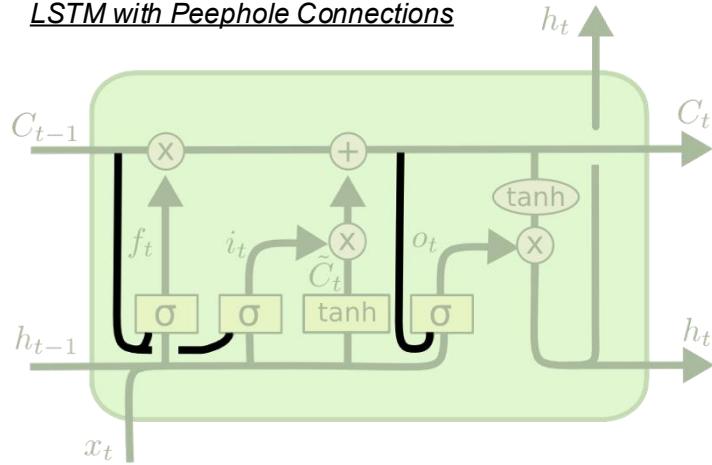


- Can a network can learn to extract relevant information conveyed by the size of the time lags?
 - LSTMs with peephole connections
- Are all connections in an LSTM cell necessary?
 - LSTMs with coupled input and forget gates

LSTM with Peephole Connections



LSTM with Peephole Connections



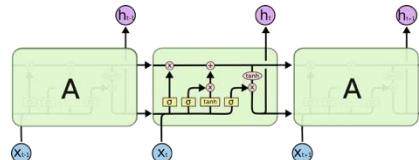
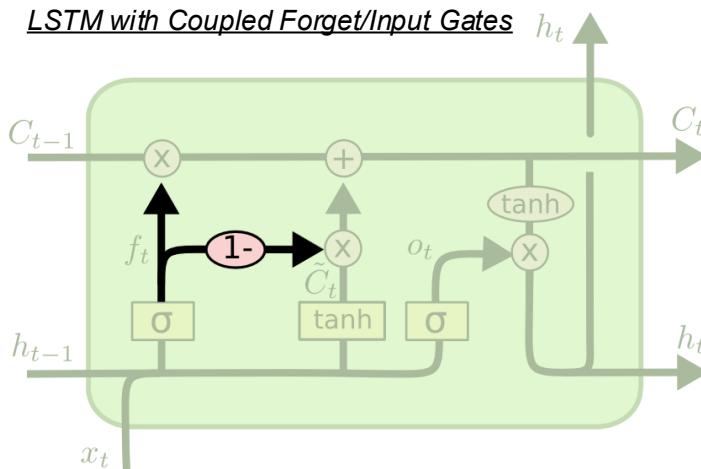
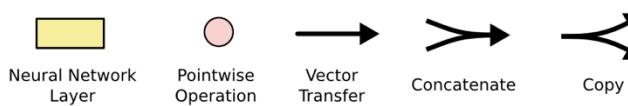
Idea: allow gate layers to look at the cell state

$$f_t = \sigma(W_f \cdot [C_{t-1}, h_{t-1}, x_t] + b_f)$$

$$i_t = \sigma(W_i \cdot [C_{t-1}, h_{t-1}, x_t] + b_i)$$

$$o_t = \sigma(W_o \cdot [C_t, h_{t-1}, x_t] + b_o)$$

LSTM with Coupled Forget/Input Gates



Idea: instead of separately deciding what to forget and what we should add new information to, we make those decisions together

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

We only forget when we're going to input something in its place, and vice versa.



Topics for Today

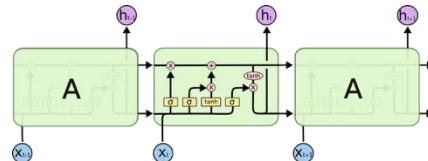
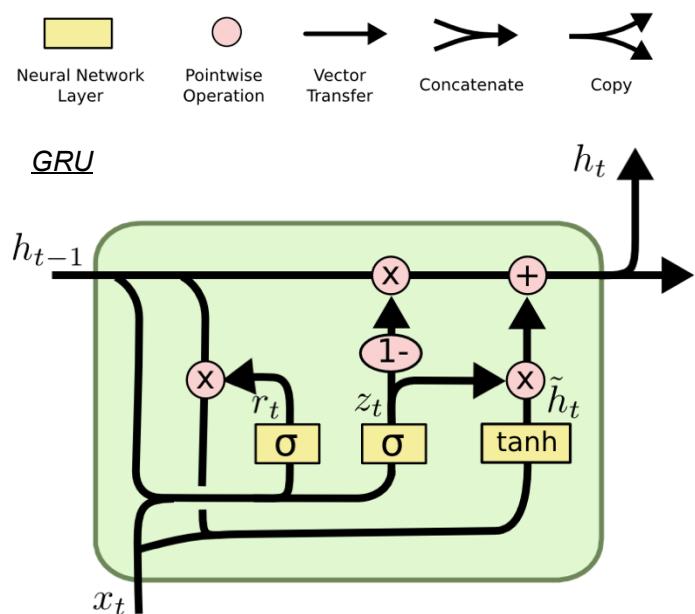
Model Architectures and Contextual Embeddings

- Long-Short Term Memory (LSTM)
- Gated Recurrent Units (GRU)
- Example Sequence Classification Tasks
- Elmo and Contextual Embeddings

Gated Recurrent Unit (GRU)

- **Reset gate:** how much of the previous state we might still want to remember
- **Update gate:** how much of the new state is just a copy of the old state.

GRU (cont.)



Idea:

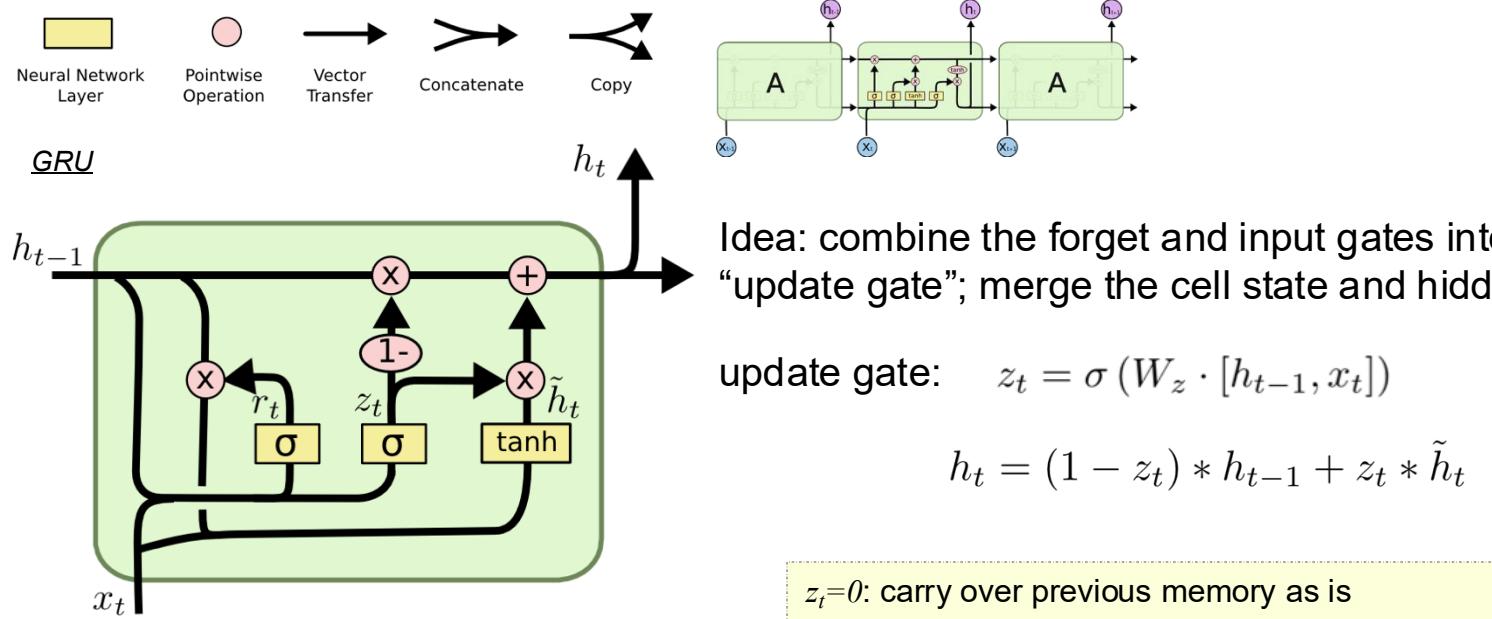
- combine the forget and input gates into a single “update gate”;
- merge the cell state and hidden state

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

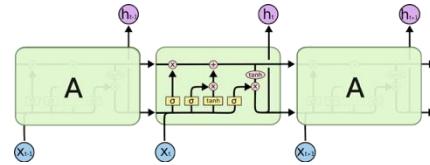
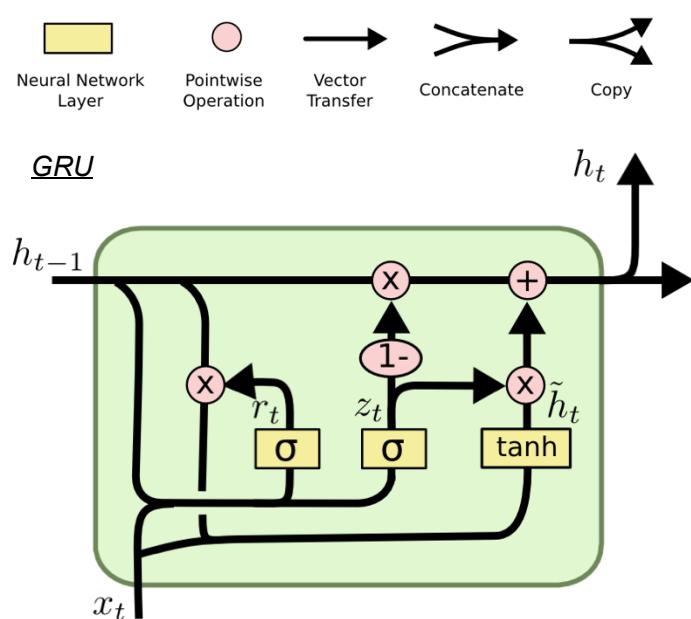
$$\tilde{h}_t = \tanh(W \cdot [r_t * h_{t-1}, x_t])$$

$r_t=0$: ignore previous memory and only store the new word information (conventional feed forward network)
 $r_t=1$: conventional RNN

GRU (cont.)



GRU (cont.)



Idea: combine the forget and input gates into a single “update gate”; merge the cell state and hidden state

update gate: $z_t = \sigma (W_z \cdot [h_{t-1}, x_t])$

reset gate: $r_t = \sigma (W_r \cdot [h_{t-1}, x_t])$

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

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

GRU is simpler and has less parameters than LSTM

LSTM and GRU Extensions



- Bi-directional RNN with LSTM/GRU cells
- Deep bi-directional RNN with LSTM/GRU cells

Pros and Cons of LSTM & GRU



Strengths:

- They aim to capture long dependencies
- They aim to solve vanishing gradient problem
- They present interpretable gating mechanisms.

Weaknesses:

- They are slow due to sequential processing
- They are memory-heavy
- Now, largely replaced by Transformers.

Further Discussion



GRUs vs. LSTMs:

- GRUs are simpler i.e., fewer gates/parameters and hence faster training and inference.
- They often perform similarly, though LSTMs can be slightly better on tasks that require very long-term dependencies.

Further Discussion



Transformers vs. LSTMs:

- Sequential processing in LSTMs, whereas attention in transformers allows parallelization (though quadratic complexity)
- LSTMs provide useful intuition for memory in sequences.
 - Emerging approach: structured state space models
 - E.g., Structured State Space Sequence Model (S4) ([Gu et al., ICLR 2022](#)) proposes a state space model (SSM) that:
 - Can capture long sequences efficiently (lengths of 10k-100k)
 - Can train in parallel like Transformers
 - Can retain the recurrence structure (good for continuous-time or streaming tasks)



Topics for Today

Model Architectures and Contextual Embeddings

- Long-Short Term Memory (LSTM)
- Gated Recurrent Units (GRU)
- Example Sequence Classification Tasks
- Elmo and Contextual Embeddings

How to Frame a Learning Problem?



- The goal of the learning algorithm f is to map the input domain X into the output domain Y

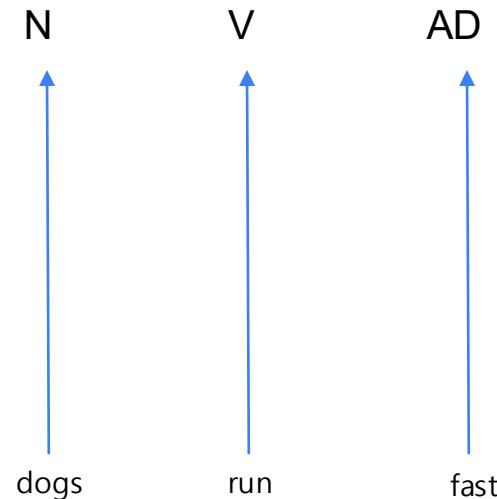
$$f : X \rightarrow Y$$

- **Input domain:** word, word sequence, audio signal, click logs
- **Output domain:** single label, sequence tags, tree structure, probability distribution

Network design should leverage input and output domain properties

POS Tagging

- Tag a word at each timestamp
 - Input: word sequence
 - Output: corresponding POS tag sequence



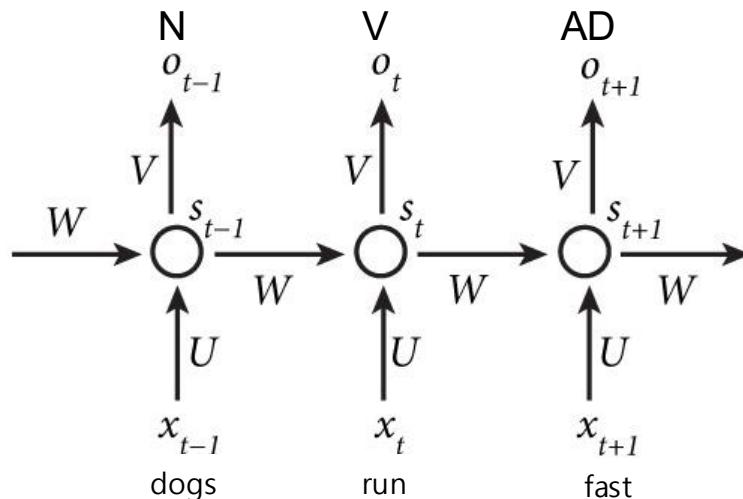
POS Tagging (cont.)

- Ambiguities: The same word can be tagged with different POS tags depending on the context.

I can can the can.

POS Tagging (cont.)

- Tag a word at each timestamp
 - Input: word sequence
 - Output: corresponding POS tag sequence



Named Entity Recognition (NER)



- Named entity: anything that can be referred to with a proper name: a person, a location, an organization
- Commonly extended to include things that aren't entities per se, including dates, times, and other kinds of temporal expressions, and even numerical expressions like prices.

Type	Tag	Sample Categories	Example sentences
People	PER	people, characters	Turing is a giant of computer science.
Organization	ORG	companies, sports teams	The IPCC warned about the cyclone.
Location	LOC	regions, mountains, seas	The Mt. Sanitas loop is in Sunshine Canyon.
Geo-Political Entity	GPE	countries, states, provinces	Palo Alto is raising the fees for parking.
Facility	FAC	bridges, buildings, airports	Consider the Golden Gate Bridge.
Vehicles	VEH	planes, trains, automobiles	It was a classic Ford Falcon.

Figure 18.1 A list of generic named entity types with the kinds of entities they refer to.

NER (cont.)



The fourth **Wells** account moving to another agency is the packaged paper-products division of **Georgia-Pacific Corp.**, which arrived at **Wells** only last fall. Like **Hertz** and **the History Channel**, it is also leaving for an Omnicom-owned agency, the **BBDO South** unit of **BBDO Worldwide**. **BBDO South** in **Atlanta**, which handles corporate advertising for **Georgia-Pacific**, will assume additional duties for brands like Angel Soft toilet tissue and Sparkle paper towels, said **Ken Haldin**, a spokesman for **Georgia-Pacific** in **Atlanta**.

PERSON **LOCATION** **ORGANIZATION**

NER (cont.)



- Finding spans of text that constitute proper name, and then classifying the type of the entity.
- Used to be a difficult task because of ambiguity of boundaries and types.

Name	Possible Categories
<i>Washington</i>	Person, Location, Political Entity, Organization, Vehicle
<i>Downing St.</i>	Location, Organization
<i>IRA</i>	Person, Organization, Monetary Instrument
<i>Louis Vuitton</i>	Person, Organization, Commercial Product

Figure 18.2 Common categorical ambiguities associated with various proper names.

[PER Washington] was born into slavery on the farm of James Burroughs.

[ORG Washington] went up 2 games to 1 in the four-game series.

Blair arrived in [LOC Washington] for what may well be his last state visit.

In June, [GPE Washington] passed a primary seatbelt law.

The [VEH Washington] had proved to be a leaky ship, every passage I made...

Figure 18.3 Examples of type ambiguities in the use of the name *Washington*.

NER as Sequence Classification

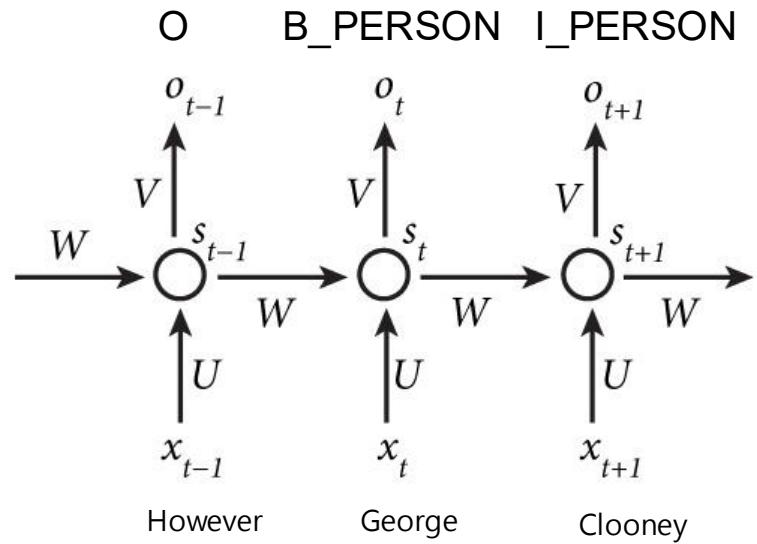
- IOB or IO representation is used for entities in sequence labeling.
- IOB representation:
 - Introduce a tag for the beginning (B) and inside (I) of each entity type, and one for tokens outside (O) any entity.
 - The number of tags is thus $2n + 1$ tags, where n is the number of entity types.
 - IOB tagging can represent exactly the same information as the bracketed notation

NER as Sequence Classification (cont.)

- Example sentence:
 - [ORG American Airlines], a unit of [ORG AMR Corp.], immediately matched the move, spokesman [PER Tim Wagner] said.

Words	IOB Label	IO Label
American	B-ORG	I-ORG
Airlines	I-ORG	I-ORG
,	O	O
a	O	O
unit	O	O
of	O	O
AMR	B-ORG	I-ORG
Corp.	I-ORG	I-ORG
,	O	O
immediately	O	O
matched	O	O
the	O	O
move	O	O
,	O	O
spokesman	O	O
Tim	B-PER	I-PER
Wagner	I-PER	I-PER
said	O	O
.	O	O

NER as Sequence Classification (cont.)





Topics for Today

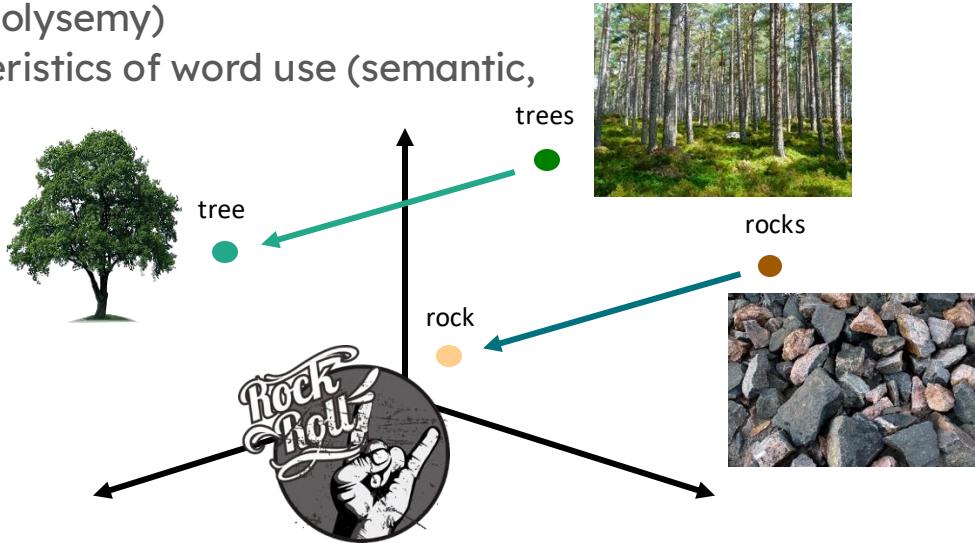
Model Architectures and Contextual Embeddings

- Long-Short Term Memory (LSTM)
- Gated Recurrent Units (GRU)
- Example Sequence Classification Tasks
- Elmo and Contextual Embeddings

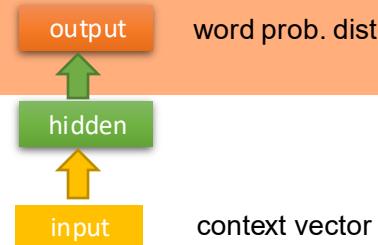
Word Embeddings: Motivating Contextual Representations



- Assumption: one representation for a word
- Issues:
 - Multiple **senses** (polysemy)
 - Complex characteristics of word use (semantic, syntactic, etc.)

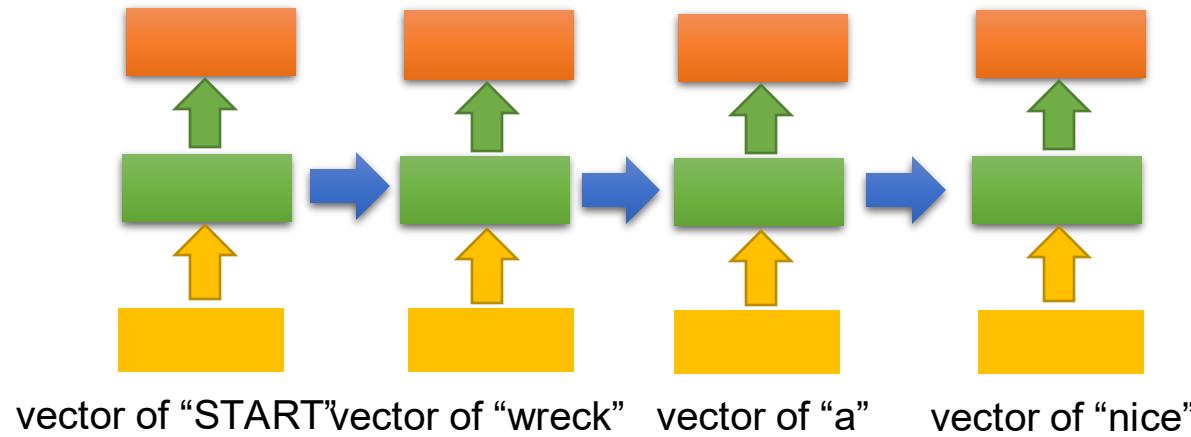


RNN-LM



- Idea: condition the neural network on all previous words and tie the weights at each time step

$P(\text{next w}=\text{"wreck"})P(\text{next w}=\text{"a"}) P(\text{next w}=\text{"nice"}) P(\text{next w}=\text{"beach"})$

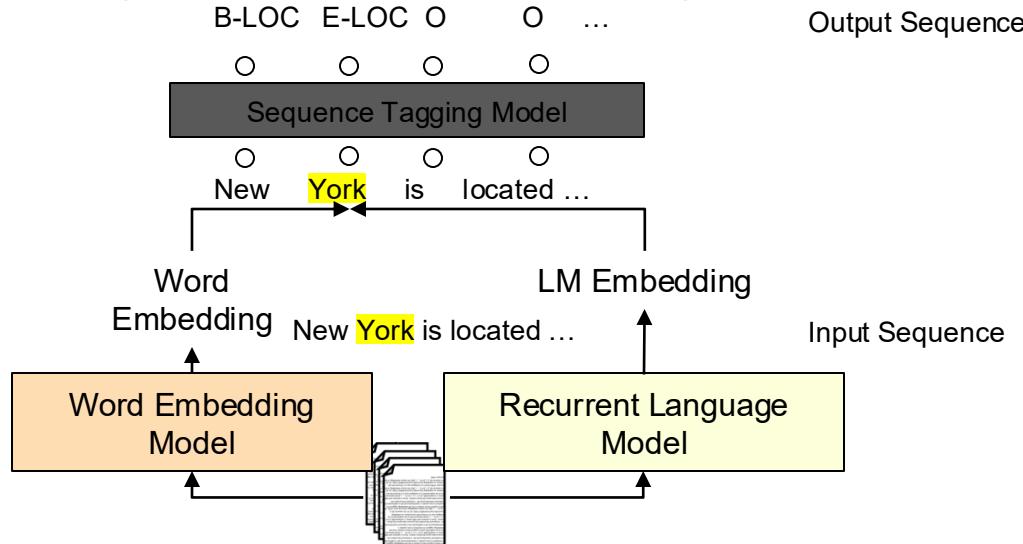


The LM produces **context-specific word representations** at each position

TagLM – “Pre-ELMo”



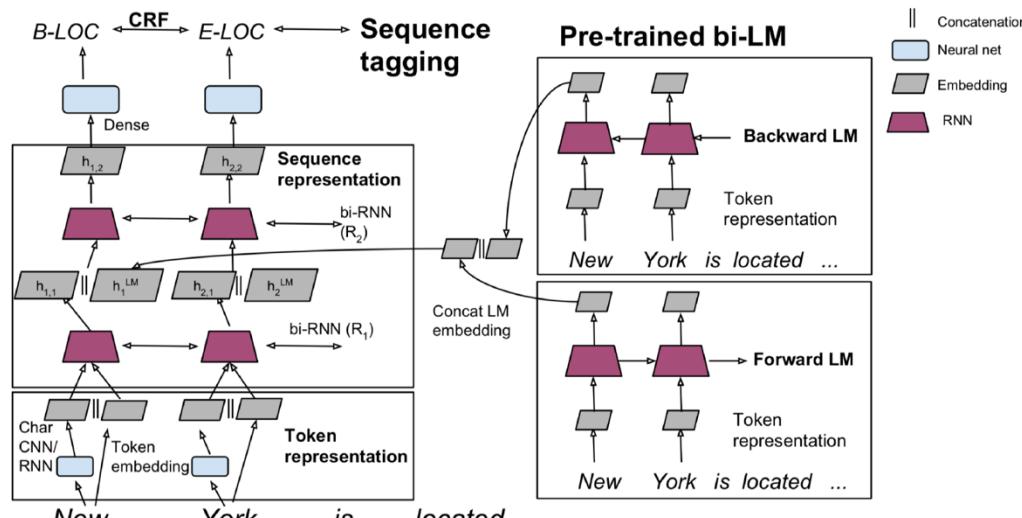
- **Idea:** train (contextual) LM on big unannotated data and provide the context-specific embeddings for the target task → **semi-supervised learning**
- Named entity recognition (NER) and chunking



[Peters et al., “Semi-supervised sequence tagging with bidirectional language models,” in ACL, 2017.](#)

TagLM - Model Details

- Leveraging pre-trained LM information



$$h_{k,1} = [\overrightarrow{h}_{k,1}; \overleftarrow{h}_{k,1}; h_k^{LM}]$$

TagLM – Steps

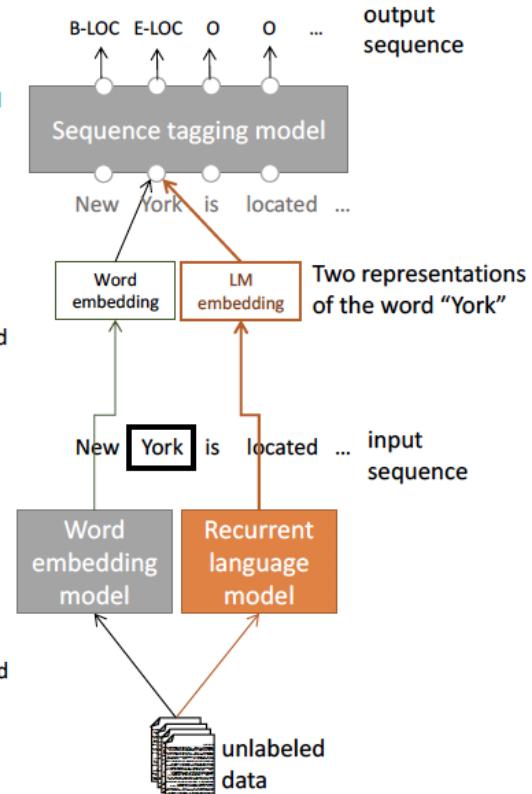


- Step 1 can use very large, unlabeled dataset.
 - Steps 2 and 3 use the training data labeled for the task.
-
- CoNLL 2003 NER English data ([Sang and De Meulder, 2003](#))
 - 1B Word Benchmark dataset ([Chelba et al, 2014](#))
 - using 800M for LM training.

English data	Articles	Sentences	Tokens
Training set	946	14,987	203,621
Development set	216	3,466	51,362
Test set	231	3,684	46,435

English data	LOC	MISC	ORG	PER
Training set	7140	3438	6321	6600
Development set	1837	922	1341	1842
Test set	1668	702	1661	1617

Step 3:
Use both word embeddings and LM embeddings in the sequence tagging model.



TagLM for Named Entity Recognition



- Find and classify names in text

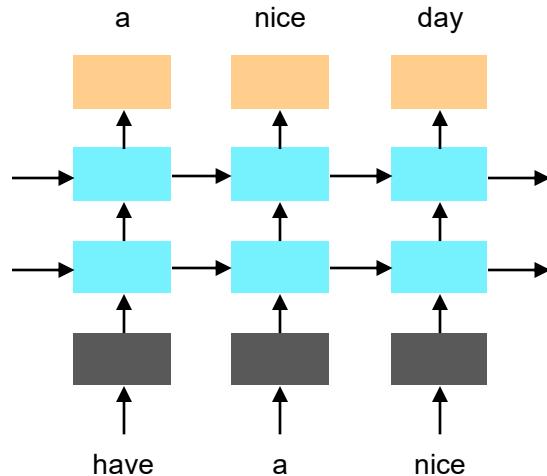
The decision by the independent MP Andrew Wilkie to withdraw his support for the minority Labor government sounded dramatic but it should not further threaten its stability. When, after the 2010 election, Wilkie, Rob Oakeshott, Tony Windsor and the Greens agreed to support Labor, they gave just two guarantees: confidence and supply.

Model	Description	CONLL 2003 F1
Klein+, 2003	MEMM softmax markov model	86.07
Florian+, 2003	Linear/softmax/TBL/HMM	88.76
Finkel+, 2005	Categorical feature CRF	86.86
Ratinov and Roth, 2009	CRF+Wiki+Word cls	90.80
Peters+, 2017	BLSTM + char CNN + CRF	90.87
Ma and Hovy, 2016	BLSTM + char CNN + CRF	91.21
TagLM (Peters+, 2017)	LSTM BiLM in BLSTM Tagger	91.93

ELMo: Embeddings from Language Models



- **Idea:** contextualized word representations
 - Learn word vectors using long contexts instead of a context window
 - Learn a deep Bi-LM and use all its layers in prediction



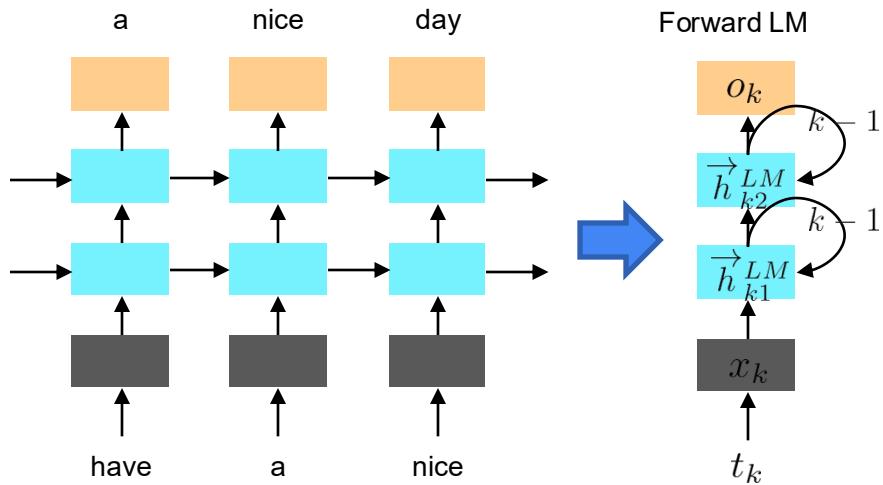
Each token is assigned a representation that is a function of entire input sentence.

ELMo: Embeddings from Language Models (cont.)



1) Bidirectional LM

$$p(t_1, t_2, \dots, t_N) = \prod_{k=1}^N p(t_k | t_1, \dots, t_{k-1})$$



ELMo: Embeddings from Language Models (cont.)



1) Bidirectional LM

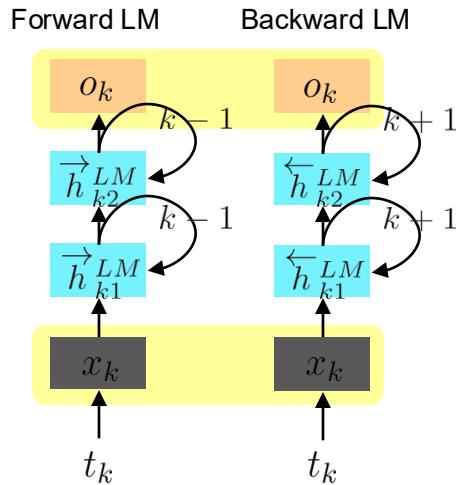
$$p(t_1, t_2, \dots, t_N) = \prod_{k=1}^N p(t_k | t_1, \dots, t_{k-1})$$

$$p(t_1, t_2, \dots, t_N) = \prod_{k=1}^N p(t_k | t_{k+1}, \dots, t_N)$$

- Character CNN for initial word embeddings
- 2 BLSTM layers

$$O = \sum_{k=1}^N \left(\log p(t_k | t_1, \dots, t_{k-1}; \Theta_x, \vec{\Theta}_{LSTM}, \Theta_s) \right.$$

$$\left. + \log p(t_k | t_{k+1}, \dots, t_N; \Theta_x, \overleftarrow{\Theta}_{LSTM}, \Theta_s) \right)$$



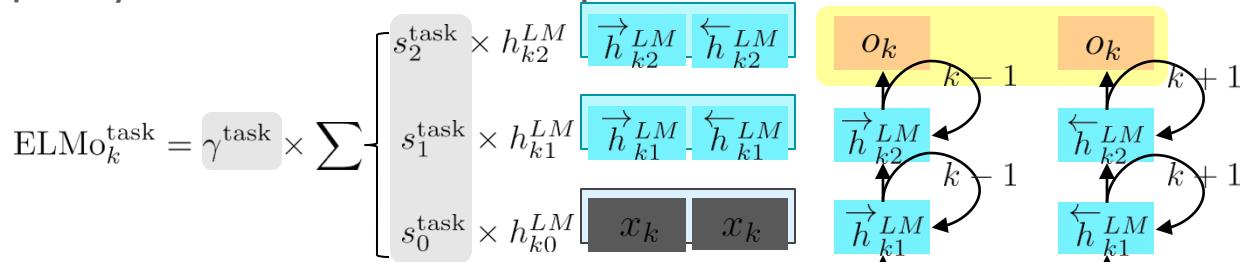
Joint maximization of log likelihood in backward and forward directions.

ELMo: Embeddings from Language Models (cont.)



2) ELMo

- Learn the task-specific linear combination of LM representations
- Use multiple layers in LSTM instead of top one



- γ^{task} scales overall usefulness of ELMo to task
- s^{task} are softmax-normalized weights
- optional layer normalization

Elmo embeddings are deep: A task-specific embedding with combination weights learned from a downstream task.

ELMo: Embeddings from Language Models (cont.)



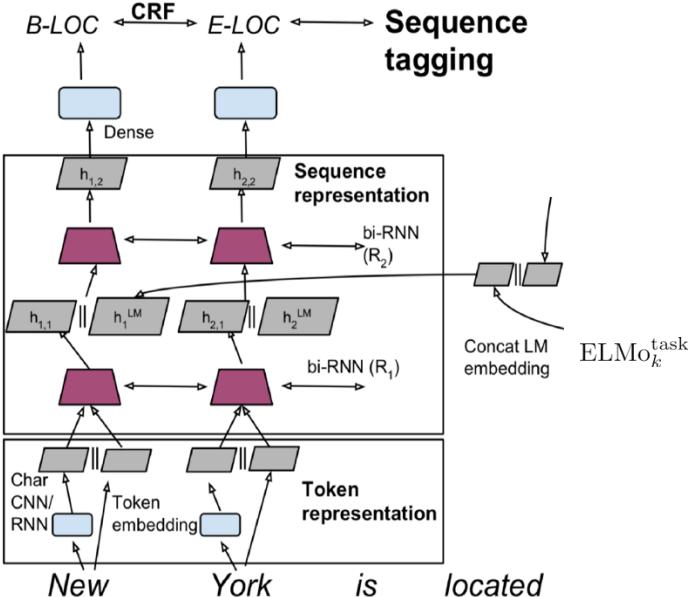
3) Use ELMo in Supervised NLP Tasks

- Get LM embedding for each word
- Freeze the LM weights and form ELMo enhanced embeddings

$[h_k; \text{ELMo}_k^{\text{task}}]$: concatenate ELMo into the intermediate layer

$[x_k; \text{ELMo}_k^{\text{task}}]$: concatenate ELMo into the input layer

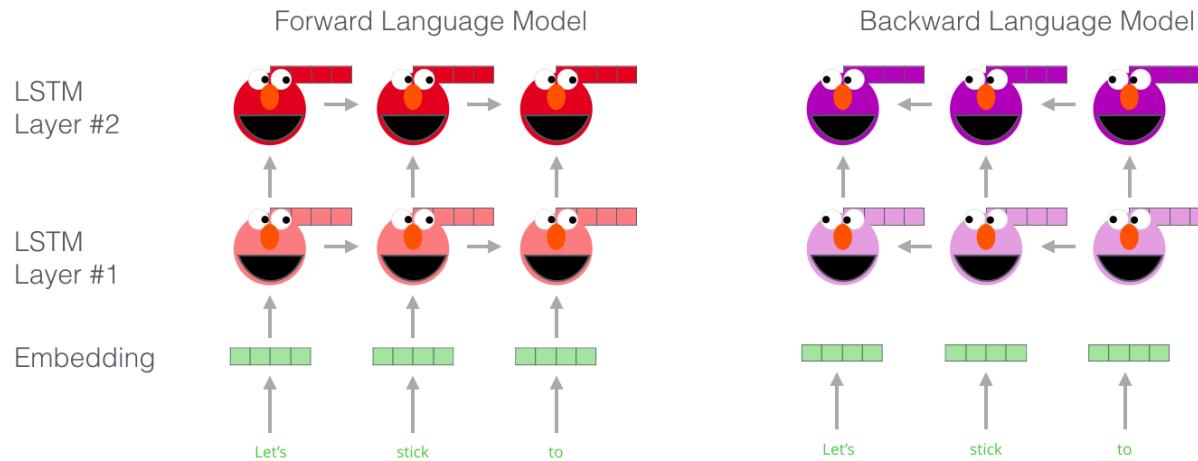
The way of concatenation depends on the task



ELMo Illustration



Embedding of “stick” in “Let’s stick to” - Step #1



Peters et al., “Deep Contextualized Word Representations”, in NAACL-HLT, 2018.

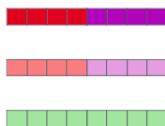
Depictions from J. Alammar’s blogpost: <https://jalammar.github.io/illustrated-bert/>

ELMo Illustration

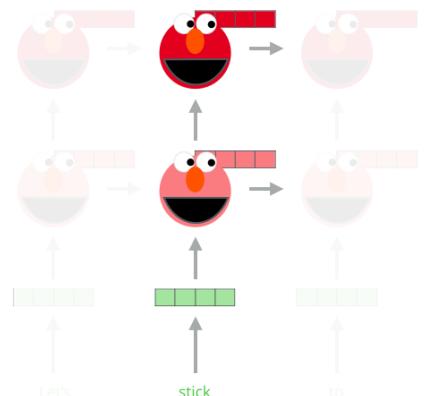


Embedding of “stick” in “Let’s stick to” - Step #2

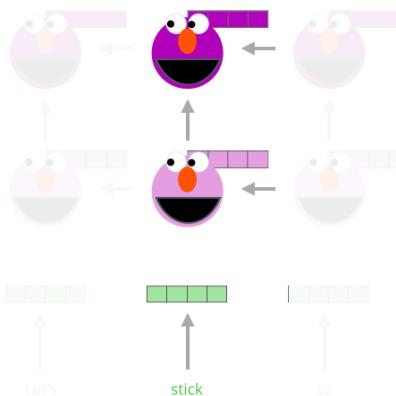
1- Concatenate hidden layers



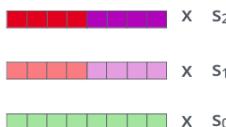
Forward Language Model



Backward Language Model



2- Multiply each vector by a weight based on the task



3- Sum the (now weighted) vectors



ELMo embedding of “stick” for this task in this context

Peters et al., “Deep Contextualized Word Representations”, in NAACL-HLT, 2018.

Depictions from J. Alammar’s blogpost: <https://jalammar.github.io/illustrated-bert/>

ELMo Performance (NLP Tasks)



- Improvement on various NLP tasks

TASK	PREVIOUS SOTA	OUR BASELINE		INCREASE (ABSOLUTE/ RELATIVE)
		ELMO + BASELINE	BASELINE	
Machine Reading Comprehension	SQuAD	Liu et al. (2017)	84.4	81.1 / 24.9%
Textual Entailment	SNLI	Chen et al. (2017)	88.6	88.0 / 5.8%
Semantic Role Labeling	SRL	He et al. (2017)	81.7	81.4 / 17.2%
Coreference Resolution	Coref	Lee et al. (2017)	67.2	67.2 / 9.8%
Name Entity Recognition	NER	Peters et al. (2017)	91.93 ± 0.19	90.15 / 21%
Sentiment Analysis	SST-5	McCann et al. (2017)	53.7	51.4 / 6.8%

Good transfer learning in NLP (similar to computer vision)

ELMo Analysis



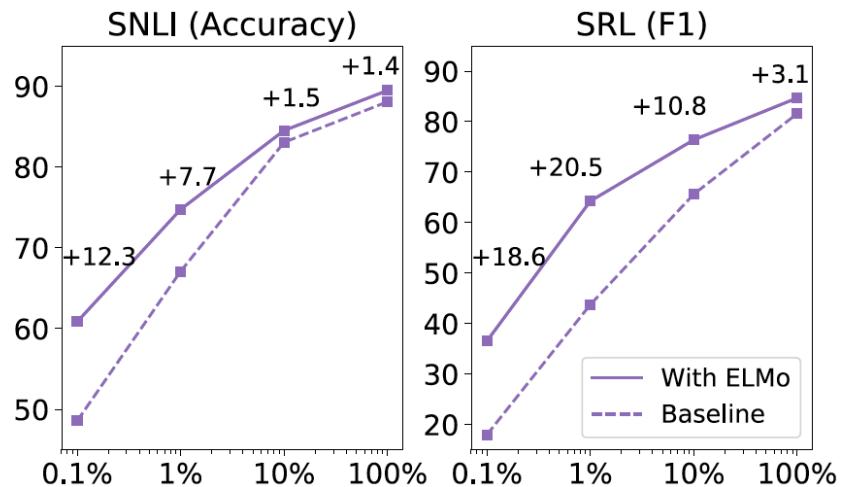
- Word embeddings vs. contextualized embeddings

Source	Nearest Neighbors
GloVe	play playing, game, games, played, players, plays, player, Play, football, multiplayer
biLM	Chico Ruiz made a spectacular <u>play</u> on Alusik 's grounder {...} Kieffer , the only junior in the group , was commended for his ability to hit in the clutch , as well as his all-round excellent play .
	Olivia De Havilland signed to do a Broadway <u>play</u> for Garson {...} {...} they were actors who had been handed fat roles in a successful <u>play</u> , and had talent enough to fill the roles competently , with nice understatement .

The biLM is able to disambiguate both the **PoS** and **word sense** in the source sentence

ELMo Analysis (cont.)

- Sample Efficiency



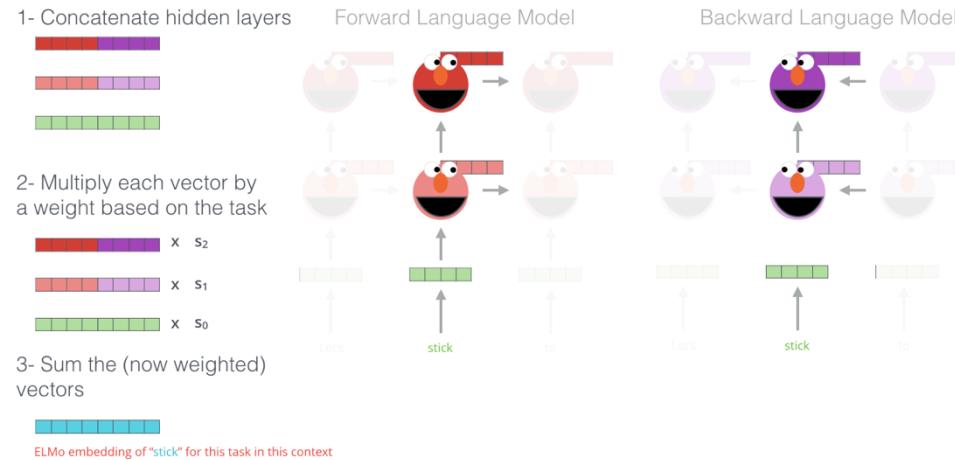
ELMo Analysis

- The two LM layers have differentiated uses/meanings
 - Lower layer is better for lower-level **syntax**, etc.
Part-of-speech tagging, syntactic dependencies, NER
 - Higher layer is better for higher-level **semantics**
Sentiment, Semantic role labeling, question answering, SNLI

PoS Tagging		Word Sense Disambiguation	
Model	Acc.	Model	F ₁
Collobert et al. (2011)	97.3	WordNet 1st Sense Baseline	65.9
Ma and Hovy (2016)	97.6	Raganato et al. (2017a)	69.9
Ling et al. (2015)	97.8	Iacobacci et al. (2016)	70.1
CoVe, First Layer	93.3	CoVe, First Layer	59.4
CoVe, Second Layer	92.8	CoVe, Second Layer	64.7
biLM, First Layer	97.3	biLM, First layer	67.4
biLM, Second Layer	96.8	biLM, Second layer	69.0

ELMo – Summary

- Contextualized embeddings learned from LMs provide informative cues
- ELMo - a general approach for learning high-quality deep context-dependent representations from biLMs
 - Pre-trained ELMo: <https://allennlp.org/elmo>
 - ELMo can process the character-level inputs



Topics for Next Week

Tuesday:

- Model Architectures, Decoding, and Attention

Thursday:

- Transformers

Midterm Information



- Not online: paper and pen
- Not open book and no cheat sheets
- Multiple choice and True/False questions only.
- From the content covered in class.

Example Question – True/False

Neural networks provide a guarantee for obtaining the global optimal solution.

- (A) TRUE (B) FALSE

Example Question – Multiple Choice

Which of the following aims to help with the vanishing gradients problem of recurrent neural networks?

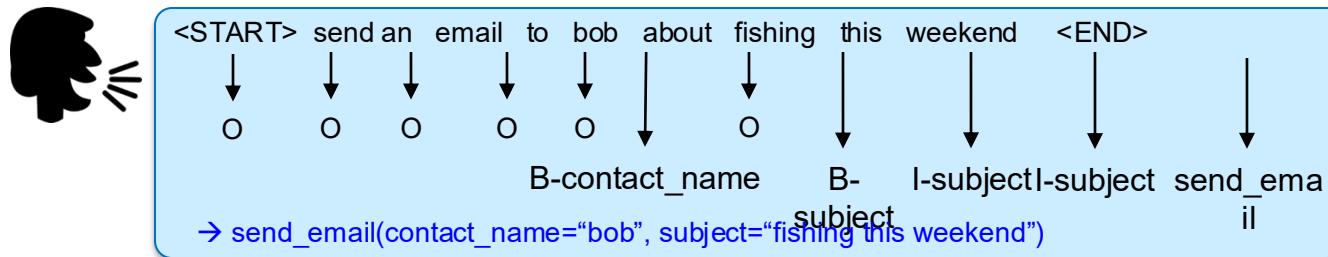
- (A) Gradient clipping
- (B) Gating mechanisms capturing longer term dependencies
- (C) Weight decay
- (D) A and B
- (E) None of the above

Main Topics for Today

1. Contextual Embeddings
2. **Pre-train** embeddings using **large unlabeled** datasets. Two strategies for applying pre-trained language representations to downstream tasks:
 - **Feature-based approach**, uses task-specific architectures that include the pre-trained representations as additional features (i.e., ELMo (Peters et al., 2018)).
 - **Fine-tuning approach**, introduces minimal task-specific parameters, and is trained on the downstream tasks by simply fine-tuning all pretrained parameters, using **smaller** datasets with task-specific labels (i.e., the Generative Pre-trained Transformer (OpenAI GPT) (Radford et al., 2018) or BERT (Devlin et al, 2019)).

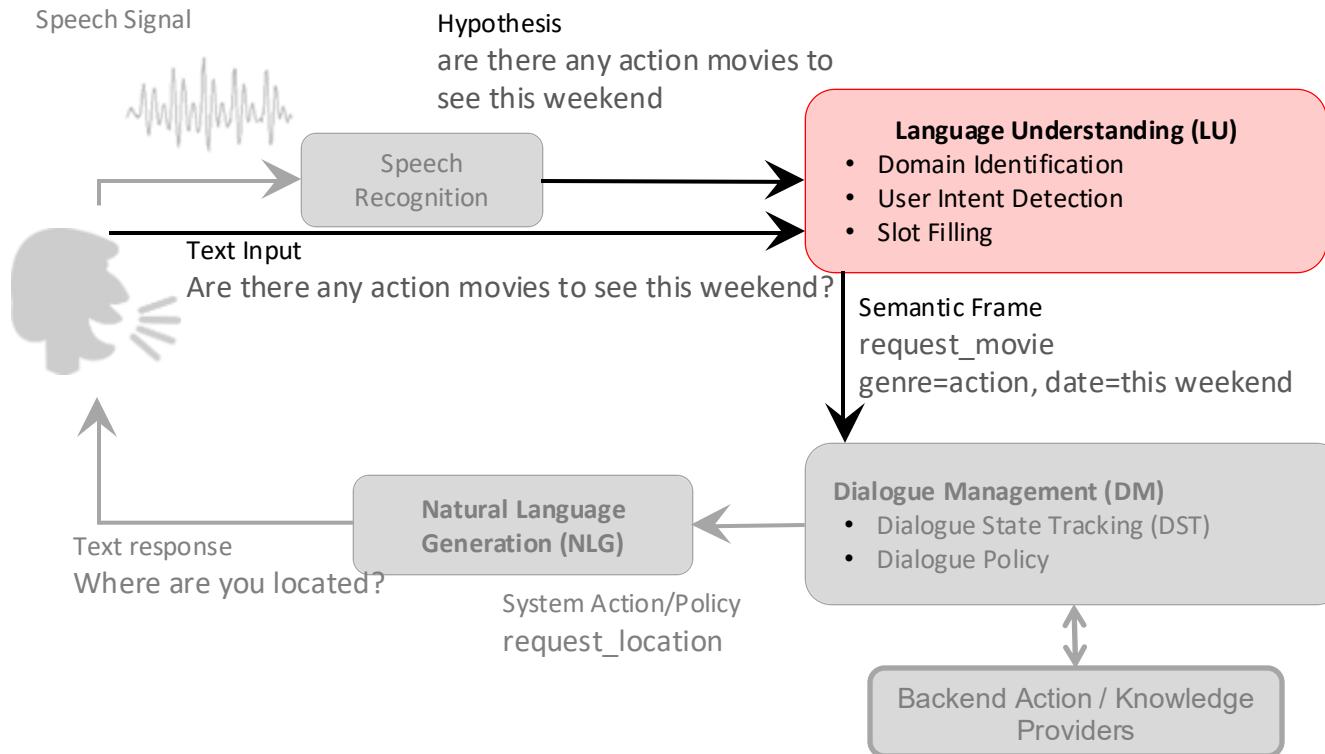
Natural Language Understanding (NLU)

- Tag a word at each timestamp
 - Input: word sequence
 - Output: IOB-format slot tag and intent tag



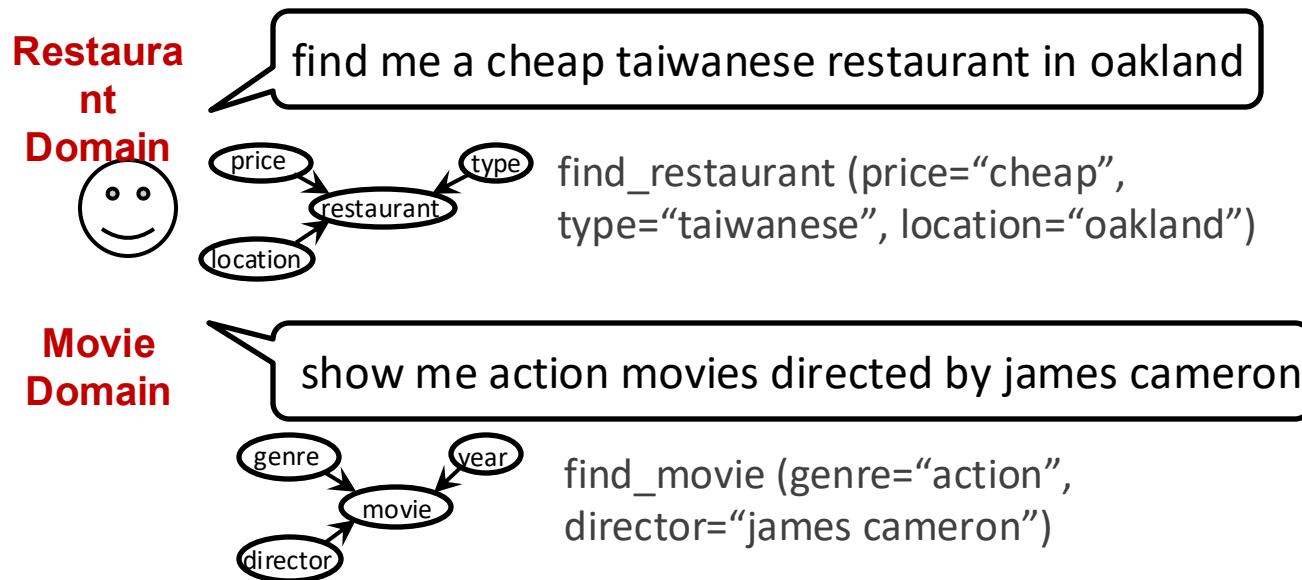
Temporal orders for input and output are the same

Task-Oriented Dialogue Systems (Young, 2000)



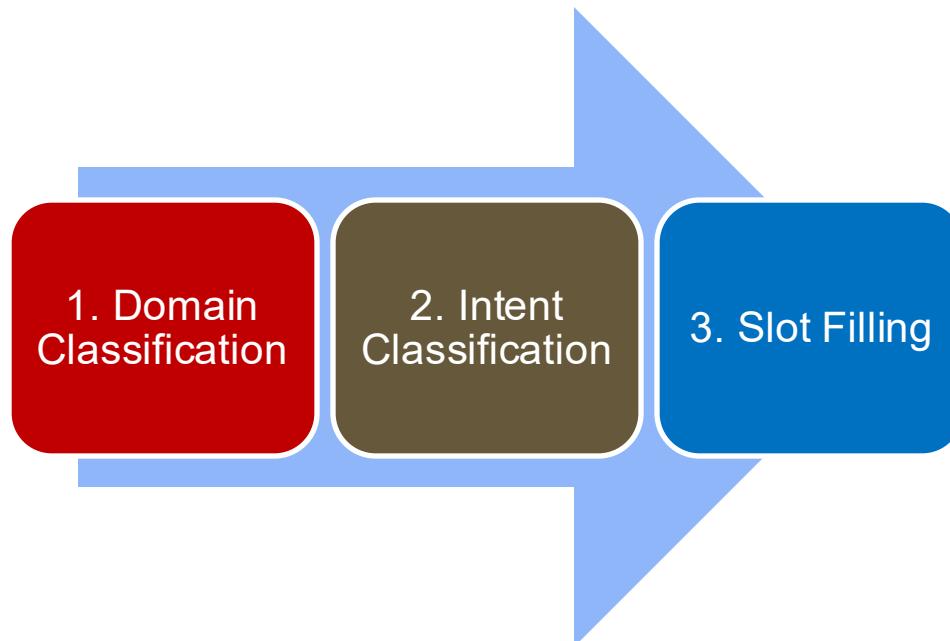
Semantic Frame Representation

- Requires a domain ontology: early connection to **backend**
- Contains **core content (intent, a set of slots with fillers)**



Language Understanding (LU)

- Pipelined



LU - Domain/Intent Classification

- Given a collection of utterances u_i with labels c_i ,
 $D = \{(u_1, c_1), \dots, (u_n, c_n)\}$ where
 $c_i \in C$, train a model to estimate labels for new

As an utterance classification task

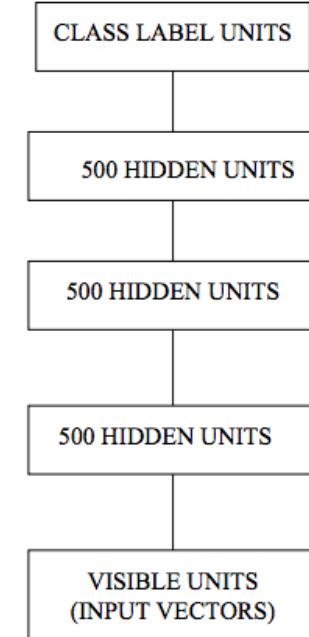
find me a cheap taiwanese restaurant in oakland

Movies	find_movie, buy_tickets
Restaurants	find_restaurant, find_price, book_table
Music	find_lyrics, find_singer
Sports	...
...	
Domain	Intent

Deep Neural Networks for Domain/Intent Classification – I (Sarikaya et al, 2011)

<http://ieeexplore.ieee.org/abstract/document/5947649/>

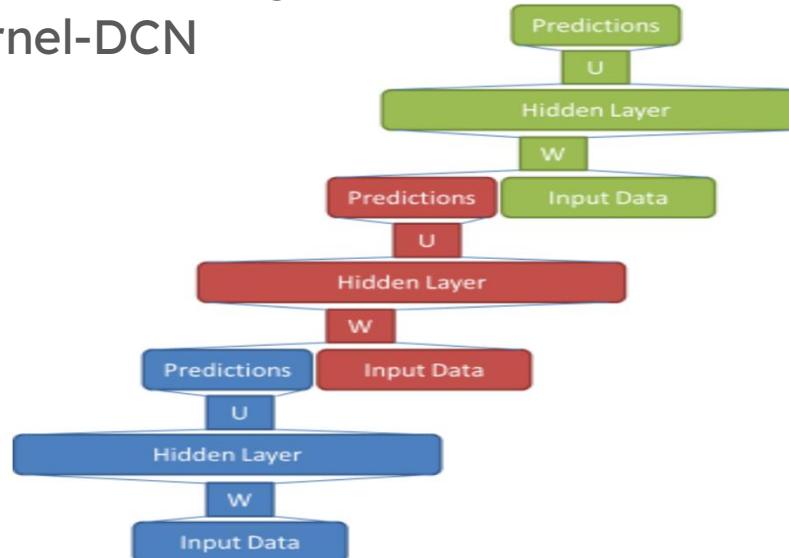
- Deep belief nets (DBN)
 - Unsupervised training of weights
 - Fine-tuning by back-propagation
 - Compared to MaxEnt, SVM, and boosting



Deep Neural Networks for Domain/Intent Classification – II (Tur et al., 2012; Deng et al., 2012)

<http://ieeexplore.ieee.org/abstract/document/6289054/>; <http://ieeexplore.ieee.org/abstract/document/6424224/>

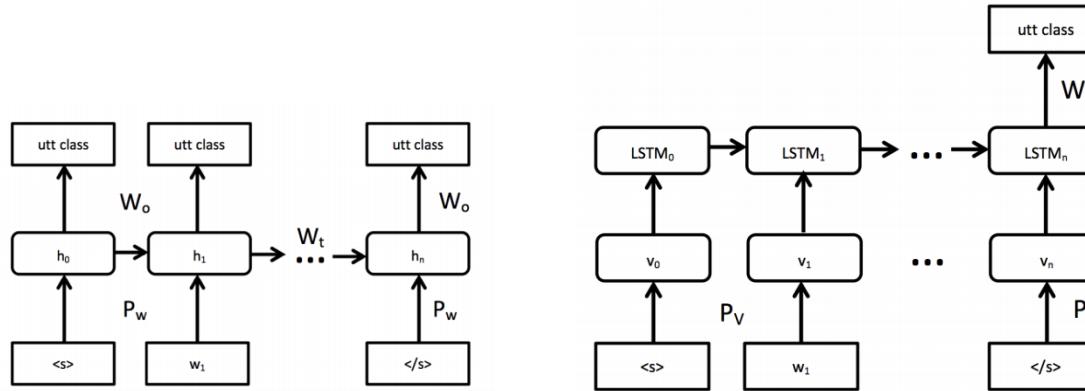
- Deep convex networks (DCN)
 - Simple classifiers are stacked to learn complex functions
 - Feature selection of salient n-grams
- Extension to kernel-DCN



Deep Neural Networks for Domain/Intent Classification – III (Ravuri & Stolcke, 2015)

https://www.microsoft.com/en-us/research/wp-content/uploads/2016/02/RNNLM_addressee.pdf

- RNN and LSTMs for utterance classification

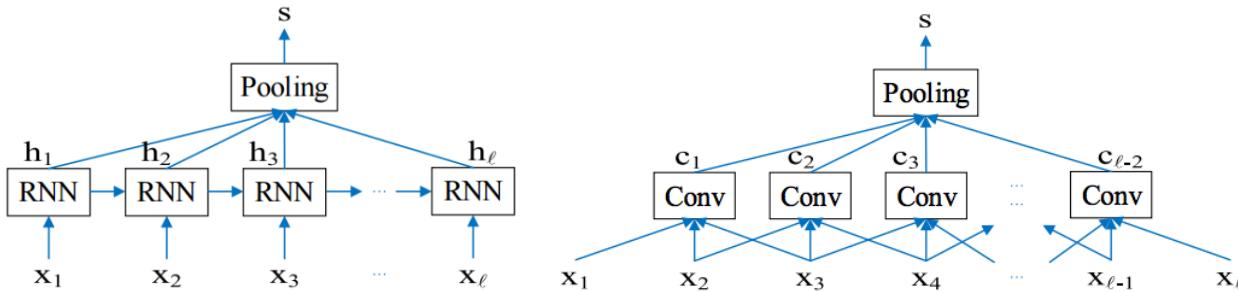


Intent decision after reading all words performs better

Deep Neural Networks for Dialogue Act Classification

- IV (Lee & Dernoncourt, 2016)

- RNN and CNNs for dialogue act classification



LU – Slot Filling

As a sequence tagging task

Given a collection tagged word sequences, $S = \{((w_{1,1}, w_{1,2}, \dots, w_{1,n_1}), (t_{1,1}, t_{1,2}, \dots, t_{1,n_1})), ((w_{2,1}, w_{2,2}, \dots, w_{2,n_2}), (t_{2,1}, t_{2,2}, \dots, t_{2,n_2})) \dots\}$

where $t_i \in M$, the goal is to estimate tags for a new word

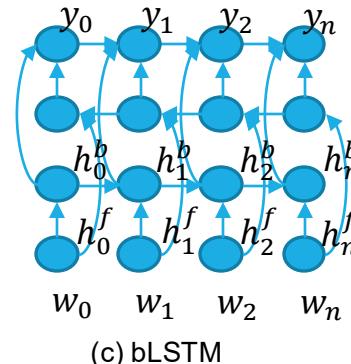
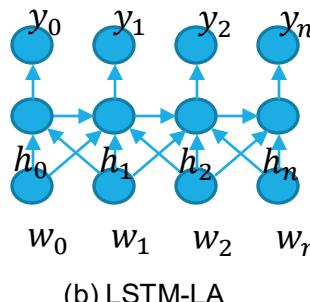
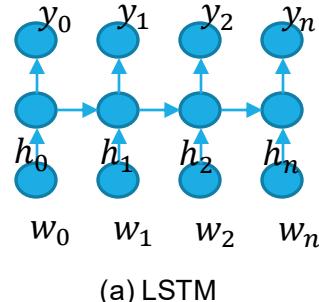
flights from Boston to New York today

	flights	from	Bosto n	to	New	York	today
Entity							
Tag	O	O	B-city	O	B-city	I-city	O
Slot	O	O	B-dept	O	B-arrival	I-arrival	B-date
Tag							

Recurrent Neural Nets for Slot Tagging – I (Yao et al, 2013; Mesnil et al, 2015)

<http://131.107.65.14/en-us/um/people/gzweig/Pubs/Interspeech2013RNNLU.pdf>; <http://dl.acm.org/citation.cfm?id=2876380>

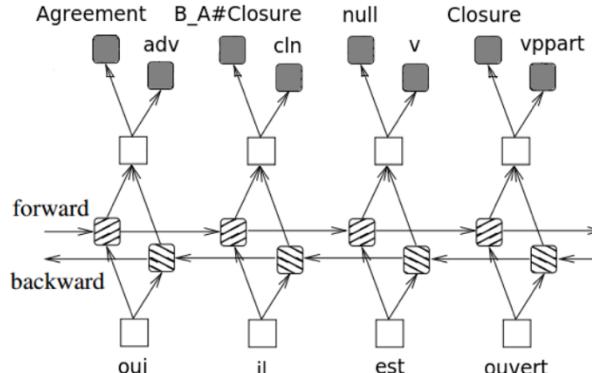
- Variations:
 - a. RNNs with LSTM cells
 - b. Input, sliding window of n-grams
 - c. Bi-directional LSTMs



Recurrent Neural Nets for Slot Tagging – III (Jaech et al., 2016; Tafforeau et al., 2016)

<https://arxiv.org/abs/1604.00117>; http://www.sensei-conversation.eu/wp-content/uploads/2016/11/favre_is2016b.pdf

- Multi-task learning
 - Goal: exploit data from domains/tasks with a lot of data to improve ones with less data
 - Lower layers are shared across domains/tasks
 - Output layer is specific to task

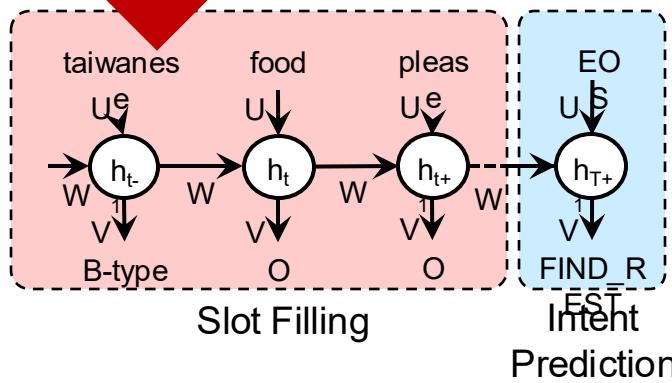


Joint Semantic Frame Parsing

https://www.microsoft.com/en-us/research/wp-content/uploads/2016/06/IS16_MultiJoint.pdf; <https://arxiv.org/abs/1609.01454>

Sequence-based
(Hakkani-Tur et al., 2016)

- Slot filling and intent prediction in the same output sequence



NLU Evaluation

- Domain and Intent Classification -> F-measure, similar to homework 1
- Slot Tagging: Slot F-measure
 - Both the slot span and category should be estimated correctly.

		flights	from	Bosto n	to	New	York	at	5pm	today	
Actual	Predicted	O	O	B-dept	O	B-arrival	I- arrival	O	B-time	E-date	
	B_time	O	B-dept	O	B-arrival	O	O	B_#ppl	O		
False Positive & False Negative										False Positive & False Negative	
False Positive										False Negative	

Homework 3

- **Training data:**

who was the main actor in the exorcist -> O O O O O O B_movie I_movie

how many woody allen movies starred diane keaton -> O O B_director
I_director O O B_cast I_cast

- **Test data:**

who played in the movie captain America -> ?

- **Aim:** Training vanilla RNNs, RNNs with LSTM cells and so on...

Homework 3 (cont.)

- 2,312 utterances in the training set (subset of HW1)
- 981 utterances in the test set (same as HW1)
- 13 slot types in the training set (27 possible IOB tags): Cast, char, country, director, genre, language, location, movie, mpaa_rating, person, producer, release_year, subject

Suggestions for Homework 3

- You'll need to change how labels are obtained from the output layer
- Study the examples in the book and others available on the web
- Start implementing simpler models (i.e., vanilla RNN)
- Then move towards more complex ones (LSTM, etc.)

Suggestions for Homework 3

- Dynamic RNN (unroll for each example) OR
- Fixed size utterances (chop long utterances and pad short ones), for example, if fixed size is 8:
 - I want to book a table at Cascal for 8 people tomorrow -> I want to book a table at Cascal
 - Will miss date=tomorrow, #_people =8
 - 2 people -> 2 people PAD PAD PAD PAD PAD PAD
 - Many unnecessary computations
- Best system from last year: multiple LSTM layers + a CRF layer on the top to ensure output label consistency!

Wednesday

- Midterm in classroom (today's content not included!)
- Starts at 5:20pm sharp, until 6:50pm!

Outline for Today

- Long Short Term Memory (LSTM)
 - Implementing LSTMs
 - Gated Recurrent Units (GRU)
 - Sequence Classification Tasks and homework 3
- Reading:
 - Starting [Chapter 9 of Dive Into Deep Learning](#)
 - [Chapter 7 of NLP with PyTorch](#)

Implementing LSTMs from scratch

- Code by Varuna Jayasiri, using numpy:
https://blog.varunajayasiri.com/numpy_lstm.html
- Inspired by Karpathy's character LM

Implementing LSTMs from scratch (cont.)

```
def forward(x, h_prev, C_prev, p = parameters):
    assert x.shape == (X_size, 1)
    assert h_prev.shape == (H_size, 1)
    assert C_prev.shape == (H_size, 1)

    z = np.row_stack((h_prev, x))
    f = sigmoid(np.dot(p.W_f.v, z) + p.b_f.v)
    i = sigmoid(np.dot(p.W_i.v, z) + p.b_i.v)
    C_bar = tanh(np.dot(p.W_C.v, z) + p.b_C.v)

    C = f * C_prev + i * C_bar
    o = sigmoid(np.dot(p.W_o.v, z) + p.b_o.v)
    h = o * tanh(C)

    v = np.dot(p.W_v.v, h) + p.b_v.v
    y = np.exp(v) / np.sum(np.exp(v)) #softmax

    return z, f, i, C_bar, C, o, h, v, y
```

$$f_t = \sigma(W_f \cdot [h_{t-1}, x_t] + b_f)$$

$$i_t = \sigma(W_i \cdot [h_{t-1}, x_t] + b_i)$$

$$\tilde{C}_t = \tanh(W_C \cdot [h_{t-1}, x_t] + b_C)$$

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

$$o_t = \sigma(W_o \cdot [h_{t-1}, x_t] + b_o)$$

$$h_t = o_t * \tanh(C_t)$$

Implementing LSTMs from scratch (cont.)

```
In [9]: def backward(target, dh_next, dC_next, C_prev,
                  z, f, i, C_bar, C, o, h, v, y,
                  p = parameters):

    assert z.shape == (X_size + H_size, 1)
    assert v.shape == (X_size, 1)
    assert y.shape == (X_size, 1)

    for param in [dh_next, dC_next, C_prev, f, i, C_bar, C, o, h]:
        assert param.shape == (H_size, 1)

    dv = np.copy(y)
    dv[target] -= 1

    p.W_v.d += np.dot(dv, h.T)
    p.b_v.d += dv

    dh = np.dot(p.W_v.v.T, dv)
    dh += dh_next
    do = dh * tanh(C)
    do = dsigmoid(o) * do
    p.W_o.d += np.dot(do, z.T)
    p.b_o.d += do
    ...
```