Deep Learning
For Natural Language Processing

Presented By: Quan Wan, Ellen Wu, Dongming Lei

University of Illinois at Urbana-Champaign

Introduction to NLP

- Introduction to Natural Language Processing
- Word Representation
- Language Model
- Question Answering
- Coreference Resolution
- Syntactic Parsing (Dependency & Constituency)
- Conclusion

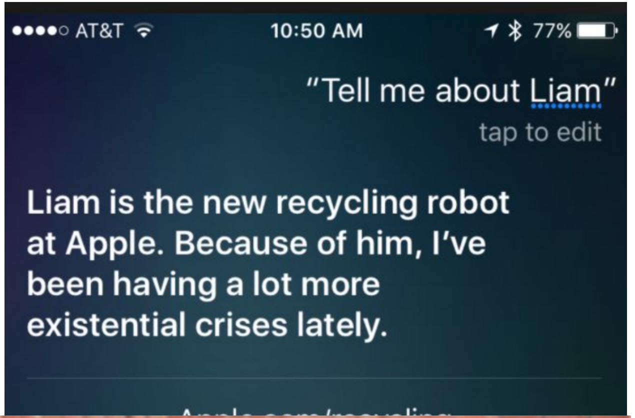
Introduction to NLP

What is natural language processing?

Difficult?

Where do we use natural language processing?

- Question answering
- Machine translation
- A lot More !



http://blog.webcertain.com/machine-translation-technology-the-search-engine-takeover/18/02/2015/

https://sixcolors.com/post/2016/04/siri-tells-you-all-about-liam/

Introduction to NLP

So NLP is something that can help machines achieve these tasks, right?

We can define NLP as:

- A work which enables machines to "understand" human language and further performs useful tasks
- It needs knowledge from CS, AI, Linguistics

Difficult!

Introduction to NLP

Difficulties in NLP:

- We omit a lot of common knowledge, which we assume the reader possesses
- We keep a lot of ambiguities, which we assume the reader knows how to resolve
    - e.g. “The man saw a boy with a telescope.”
      Who has a telescope? => Ambiguity is a killer

Introduction to NLP

Currently, what are the tools that are commonly used in NLP ?

An interesting demo here: Stanford CoreNLP Demo

- Part-Of-Speech tagging
- Entity Recognition
- Dependency Parsing
- etc

Due to the time limitation, we are gonna talk about some of these tools at the end.

Introduction to NLP

But **why** deep learning for NLP?

Most current NLP tasks work well because of human-designed features.

- Too specific and incomplete
- Require domain-specific knowledge

  => Different domain needs different features

Introduction to NLP

However, deep learning can alleviate these issues

- Features are learned automatically from examples
- The ability to capture the complicated relations

Furthermore

- Gigantic amount of data becomes available today
- Faster CPU/GPU enables us to do deep learning more efficiently

Introduction to NLP

Sounds good, right?

But how do we feed the text data into deep learning models
(e.g. the neural network) ?

This is the most basic and important step. How do we represent a word?

Word Representation

Common/intuitive way to represent a word in computer => using a vector!

A traditional approach: **discrete representation** (one-hot representation)

- Each word is represented using a vector of dimension |V| -- size of vocabulary
- "1" in one spot and "0" in all other spots

Example:

Corpus: "I like deep learning.", "I like neural networks.", "I can do NLP."

=> V = { "I", "like", "deep", "learning", "neural", "networks", "can", "do", "NLP" }

What is the one-hot representation for "like" ? (Using the above order)

=> (0, 1, 0, 0, 0, 0, 0, 0, 0)

Word Representation

<u>Problems with one-hot representation</u>

- Similar words cannot be represented in a similar way
  e.g. We have corpus with only 2 words {“skillful”, “adept”}
  vec(“skillful”) = (1,0), vec(“adept”) = (0,1)
  => The similarity is lost.

- The curse of dimensionality => computational complexity
- The vector is sparse

We need better representation !

Word Representation

Idea:
We can represent a word by utilizing the information from its other words
=> **Distributional representation**

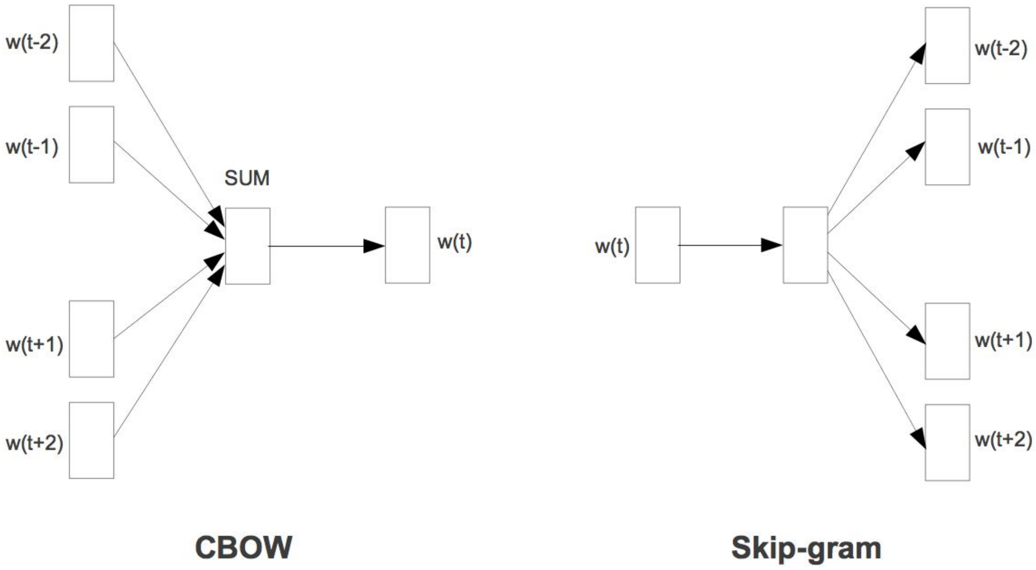
A Question:
Use **all other words** in the corpus OR just **a window** of words?
Lead to different approaches:

- Full-window approach: e.g. Latent Semantic Analysis (LSA)
- Local-window approach: e.g. Word2Vec

Word Representation

e.g. Word2Vec

- There are 2 variants -- Continuous bag-of-words (CBOW), skip-gram



Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient Estimation of Word Representations in Vector Space. ICLR, 2013.

Word Representation

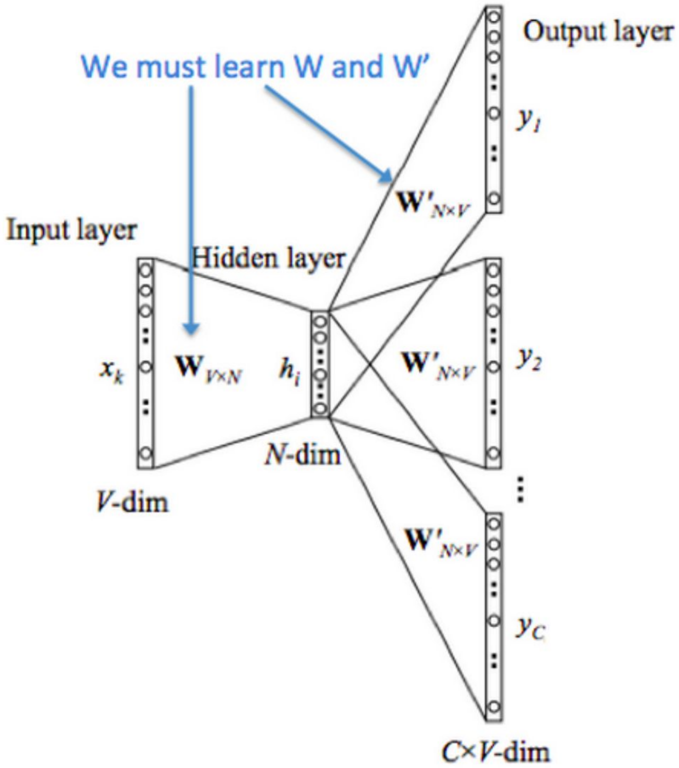
e.g. Word2Vec with skip-gram

- **<u>WI</u>**: input projection matrix of size |V|*N
- **<u>WO</u>**: output projection matrix of size N*|V|

<u>- Objective function:</u>

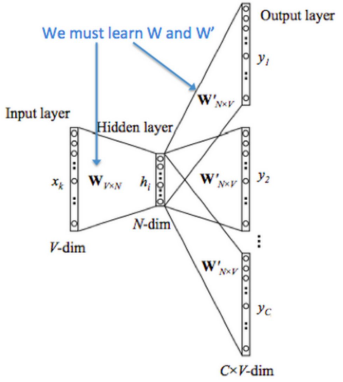
= the averaged (difference between predicted probabilistic distribution and all neighbors in the window)

<u>An example to explain!</u>



https://web.archive.org/web/20160311161826/http://cs224d.stanford.edu/lecture_notes/LectureNotes1.pdf

Word Representation



e.g. Word2Vec with skip-gram

Example:

Corpus:

"the dog saw a cat","the dog chased the cat", "The cat climbed tree"

Choose **N=3**, then:

|V| = 8, **WI** is of size 8*3, **WO** is of size 3*8

Target

The neighbors of "climbed" are: "cat", "tree"

One-hot representation:

vec("climbed") = [0 0 0 1 0 0 0 0], vec("cat") = [0 1 0 0 0 0 0 0], vec("tree") = [0 0 0 0 0 0 0 1]

Goal...

https://web.archive.org/web/20160311161826/http://cs224d.stanford.edu/lecture_notes/LectureNotes1.pdf

Word Representation

e.g. Word2Vec

Good performance in analogy test both **syntactically** and **semantically**

$$X_{car} - X_{cars} \approx X_{family} - X_{families}$$

$$X_{shirt} - X_{clothing} \approx X_{chair} - X_{furniture}$$

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient Estimation of Word Representations in Vector Space. ICLR, 2013.

Word Representation

But there are **problems**...

It **only** uses the information of a window of size N.

GloVe

Advantages:

- Leverage the global statistical information
- State-of-the-art performance on the analogy test as Word2Vec

More details at:

Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. GloVe: Global Vectors for Word Representation. EMNLP, 2014.

Language Models

What are language models?

- Language models compute the probability of occurrence of a number of words in a particular sequence. E.g. $P(w_1, \ldots, w_m)$

Why do we care about language models?

- They are useful for lots of NLP applications like machine translation, text generation and speech recognition, etc.

Language Models

Machine Translation:

- P(strong tea) > P(powerful tea)

Speech Recognition:

- P(speech recognition) > P(speech wreck ignition)

Question Answering / Summarization:

- P(President X attended …) is higher for X = Trump

…

Language Models

Conventional language models apply a fixed window size of previous words to calculate probabilities. (count-based or NN models)

$$P(w_1, ..., w_m) = \prod_{i=1}^{i=m} P(w_i | w_1, ..., w_{i-1}) \approx \prod_{i=1}^{i=m} P(w_i | w_{i-(n-1)}, ..., w_{i-1})$$

Most state-of-the-art models are based on Recurrent Neural Networks (RNN), which are capable of conditioning the model on all previous words in the corpus.

http://cs224d.stanford.edu/lecture_notes/notes4.pdf

RNN in Neural Language Model (NLM)

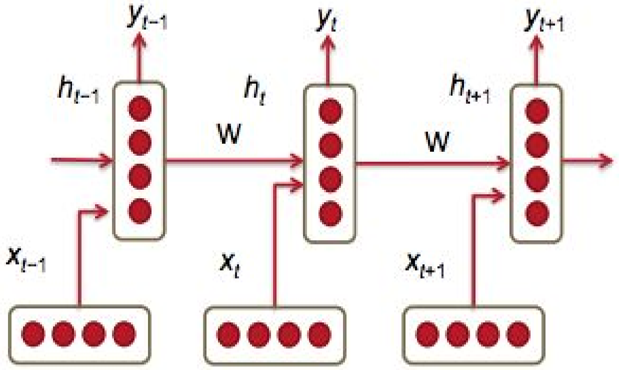
Hidden state: $$h_t = \sigma(W^{(hh)}h_{t-1} + W^{(hx)}x_{[t]})$$

Output: $$\hat{y}_t = softmax(W^{(S)}h_t)$$

Loss function at t: $$J^{(t)}(\theta) = -\sum_{j=1}^{|V|} y_{t,j} \times log(\hat{y}_{t,j})$$

The cross entropy error over a corpus of size T: $$J = -\frac{1}{T}\sum_{t=1}^{T} J^{(t)}(\theta) = -\frac{1}{T}\sum_{t=1}^{T}\sum_{j=1}^{|V|} y_{t,j} \times log(\hat{y}_{t,j})$$

A measure of confusion: $$Perplexity = 2^J$$



Three-time-step RNN

http://cs224d.stanford.edu/lecture_notes/notes4.pdf

Pointer Sentinel Mixture Models

Issues with RNN:

- RNNs for LM do best with large hidden states while hidden state is limited in capacity (parameters increase quadratically with the size of hidden state)
- Vanishing gradient still hinders learning (LSTMs capture long term dependencies … yet we only train BPTT for 35 timesteps)
- Encoding/decoding rare words is problematic

Thus, standard softmax RNNs struggle to predict rare or unseen words (OoV)!

https://arxiv.org/pdf/1609.07843.pdf

Pointer Sentinel Mixture Models

Good news:

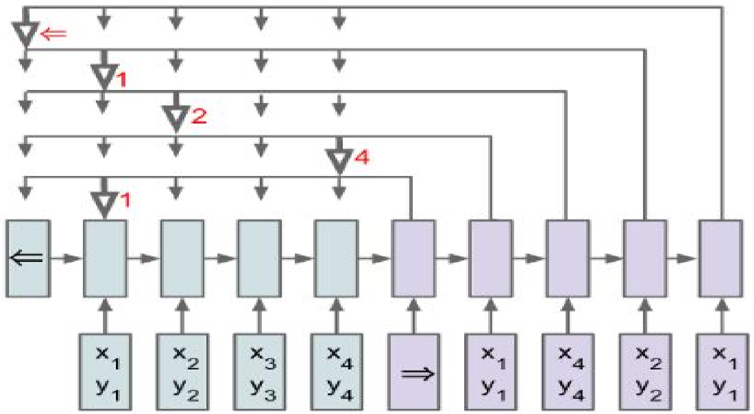
- Pointer networks (Vinyals et al., 2015) may help solve our rare / OoV problem!

How?

- A pointer network uses attention to select an element from the input as output, which allows it to produce previously unseen input tokens.

However…

- The correct answer can only be in the input with a pointer network ☹

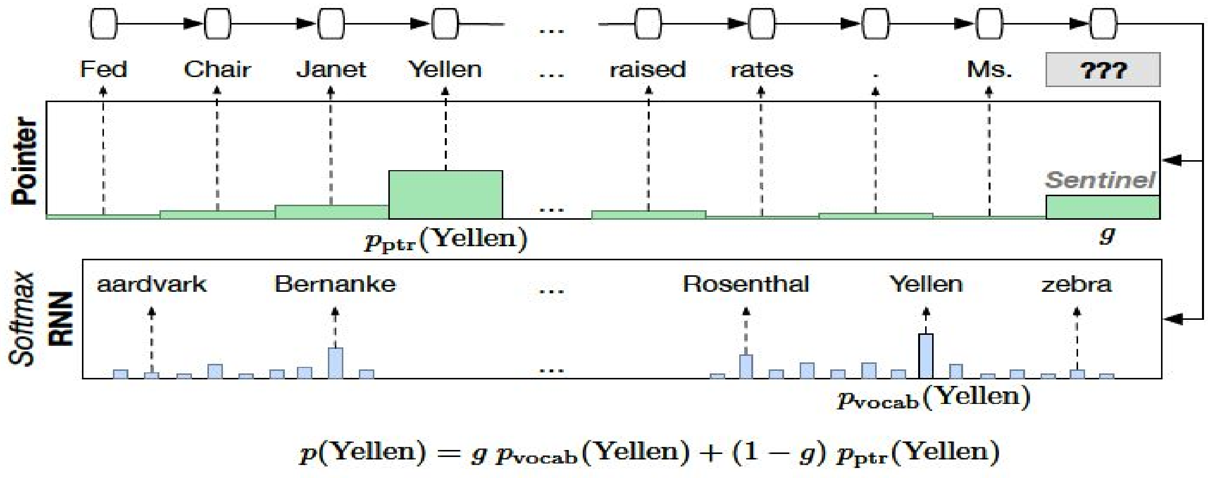


Pointer network

https://arxiv.org/pdf/1609.07843.pdf

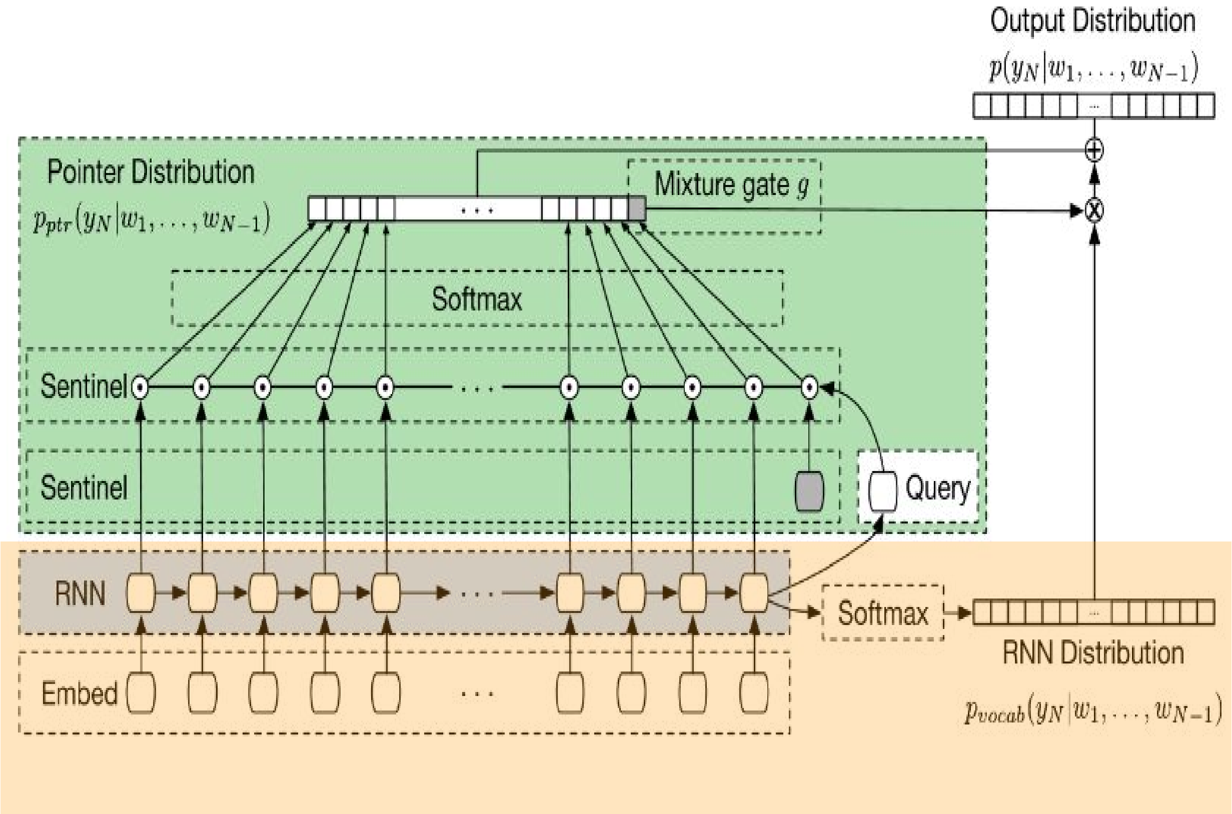
Pointer Sentinel Mixture Models

- (Merity et al., 2016) introduces a model combining vocabulary softmax (RNN) and positional softmax (a pointer component). And the pointer itself can decide how to combine through a sentinel.



https://arxiv.org/pdf/1609.07843.pdf

Pointer Sentinel Mixture Models



Softmax-RNN component:

$$p_{\text{vocab}}(w) = \text{softmax}(U h_{N-1}),$$

Pointer Network component:

$$q = \tanh(W h_{N-1} + b),$$

$$z_i = q^T h_i,$$

$$a = \text{softmax}(z)$$

$$p_{\text{ptr}}(w) = \sum_{i \in I(w,x)} a_i,$$

Mixture Model:

$$p(y_i|x_i) = g\, p_{\text{vocab}}(y_i|x_i) + (1-g)\, p_{\text{ptr}}(y_i|x_i)$$

Mixture gate:

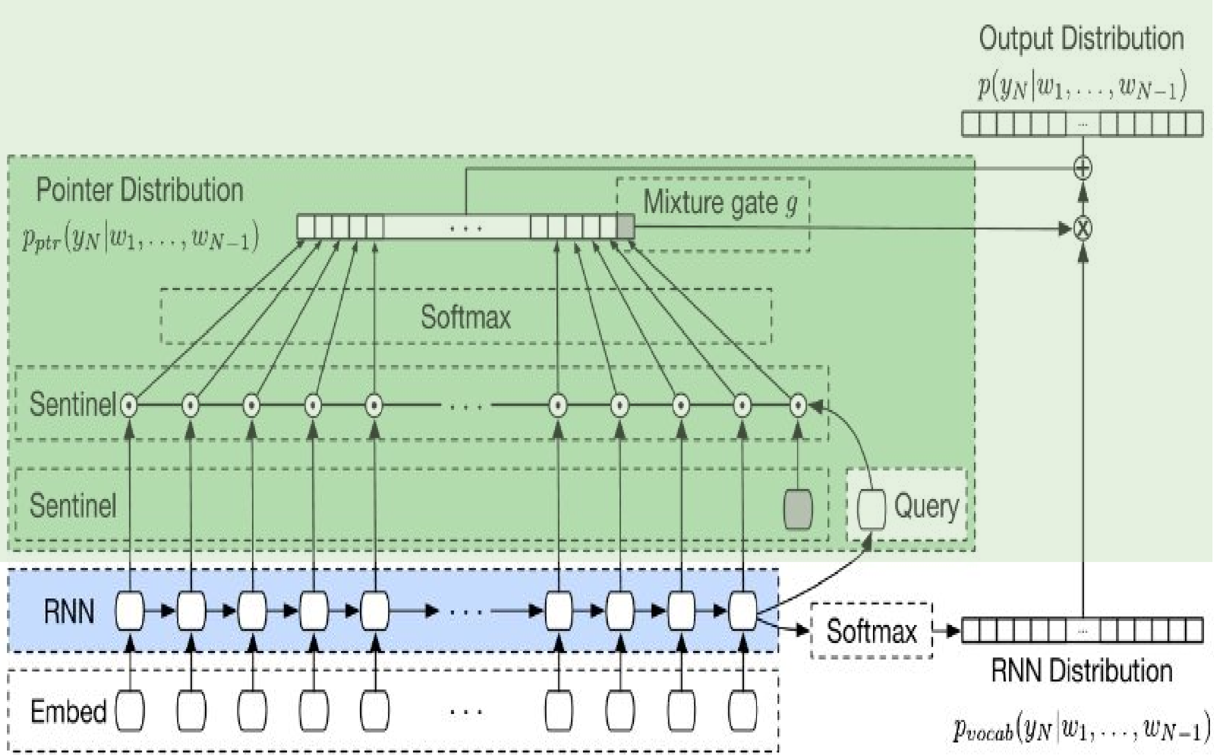
$$a = \text{softmax}\left([z; q^T s]\right)$$

$$g = a[V+1]$$

$$p_{\text{ptr}}(y_i|x_i) = \frac{1}{1-g}\, a[1:V],$$

https://arxiv.org/pdf/1609.07843.pdf

Pointer Sentinel Mixture Models



Softmax-RNN component:

$$p_{\text{vocab}}(w) = \text{softmax}(U h_{N-1}),$$

Pointer Network component:

$$q = \tanh(W h_{N-1} + b),$$

$$z_i = q^T h_i,$$

$$a = \text{softmax}(z),$$

$$p_{\text{ptr}}(w) = \sum_{i \in I(w,x)} a_i.$$

Mixture Model:

$$p(y_i|x_i) = g\, p_{\text{vocab}}(y_i|x_i) + (1-g)\, p_{\text{ptr}}(y_i|x_i)$$

Mixture gate:

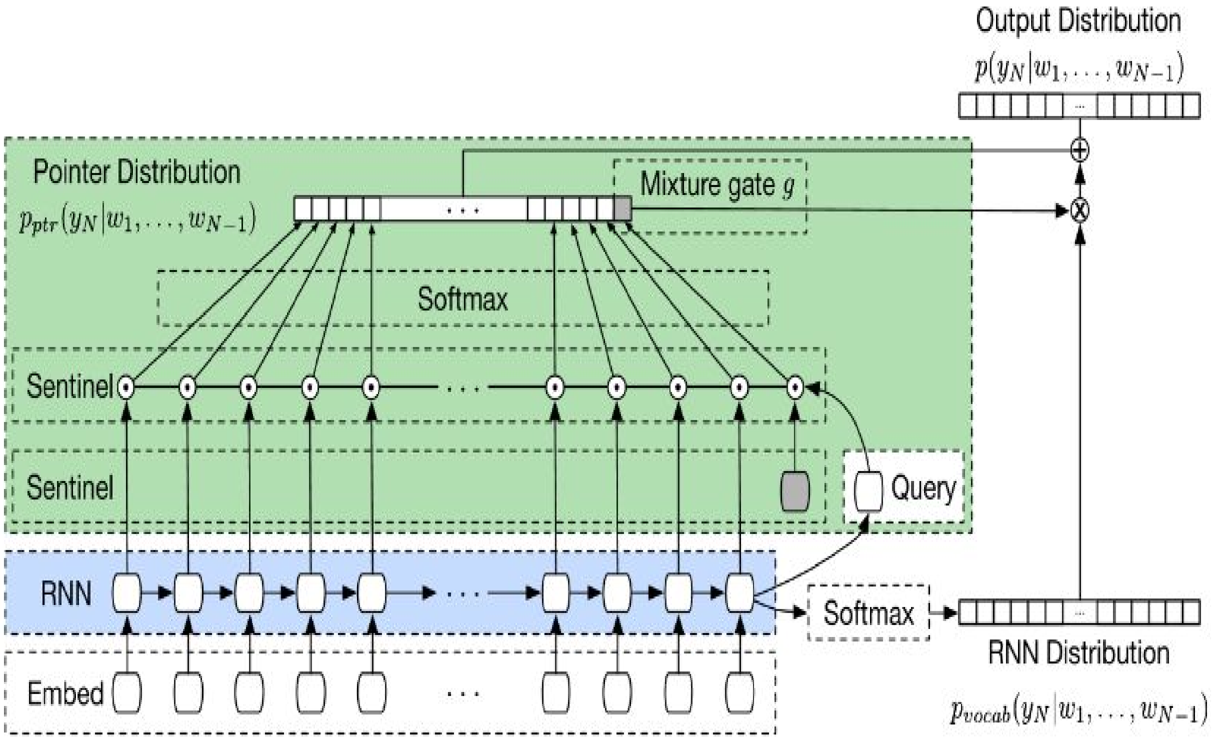
$$a = \text{softmax}\left([z; q^T s]\right)$$

$$g = a[V+1]$$

$$p_{\text{ptr}}(y_i|x_i) = \frac{1}{1-g}\, a[1:V].$$

https://arxiv.org/pdf/1609.07843.pdf

Pointer Sentinel Mixture Models



Softmax-RNN component:

$$p_{\text{vocab}}(w) = \text{softmax}(U h_{N-1}),$$

Pointer Network component:

$$q = \tanh(W h_{N-1} + b)$$

$$z_i = q^T h_i$$

$$a = \text{softmax}(z)$$

$$p_{\text{ptr}}(w) = \sum_{i \in I(w,x)} a_i$$

Mixture Model:

$$p(y_i|x_i) = g\, p_{\text{vocab}}(y_i|x_i) + (1-g)\, p_{\text{ptr}}(y_i|x_i)$$

Mixture gate:

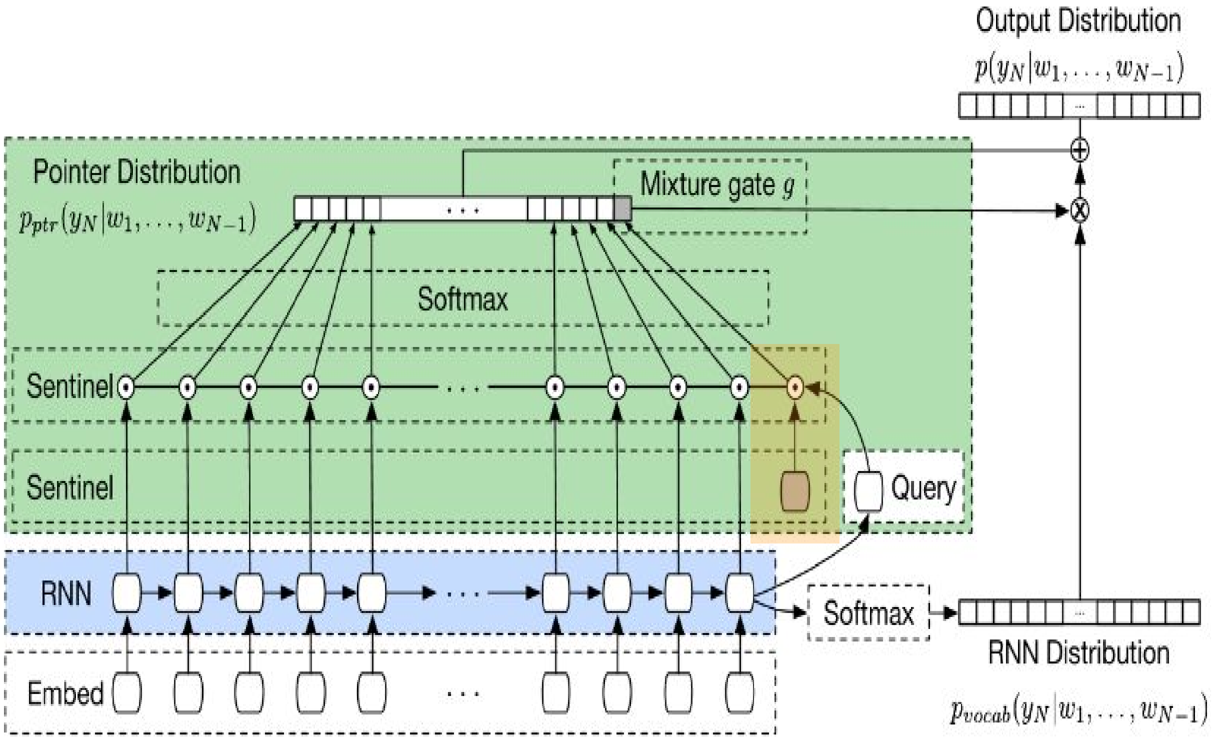
$$a = \text{softmax}\left([z; q^T s]\right)$$

$$g = a[V+1]$$

$$p_{\text{ptr}}(y_i|x_i) = \frac{1}{1-g}\, a[1:V],$$

https://arxiv.org/pdf/1609.07843.pdf

Pointer Sentinel Mixture Models



Softmax-RNN component:

$$p_{\text{vocab}}(w) = \text{softmax}(Uh_{N-1}),$$

Pointer Network component:

$$q = \tanh(Wh_{N-1} + b),$$

$$z_i = q^T h_i,$$

$$a = \text{softmax}(z),$$

$$p_{\text{ptr}}(w) = \sum_{i \in I(w,x)} a_i,$$

Mixture Model:

$$p(y_i|x_i) = g\, p_{\text{vocab}}(y_i|x_i) + (1-g)\, p_{\text{ptr}}(y_i|x_i)$$

Mixture gate:

$$a = \text{softmax}\left([z; q^T s]\right)$$

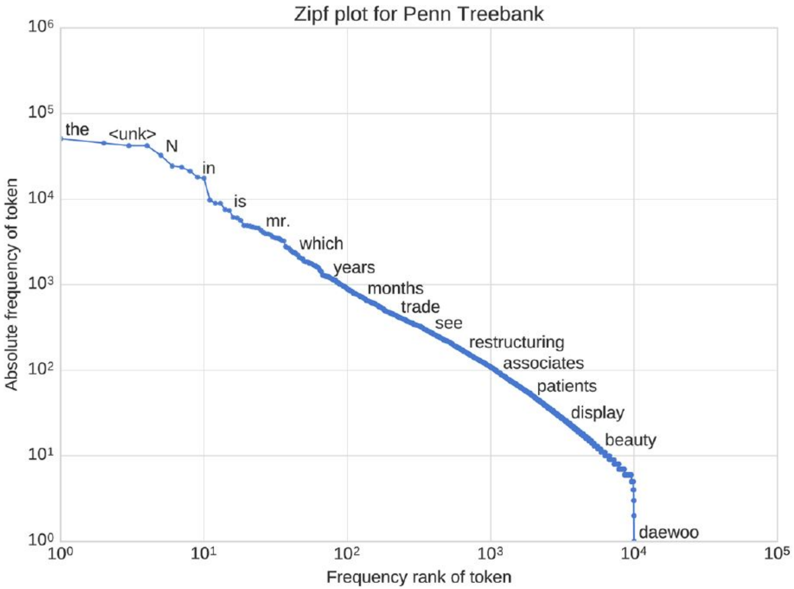
$$g = a[V+1]$$

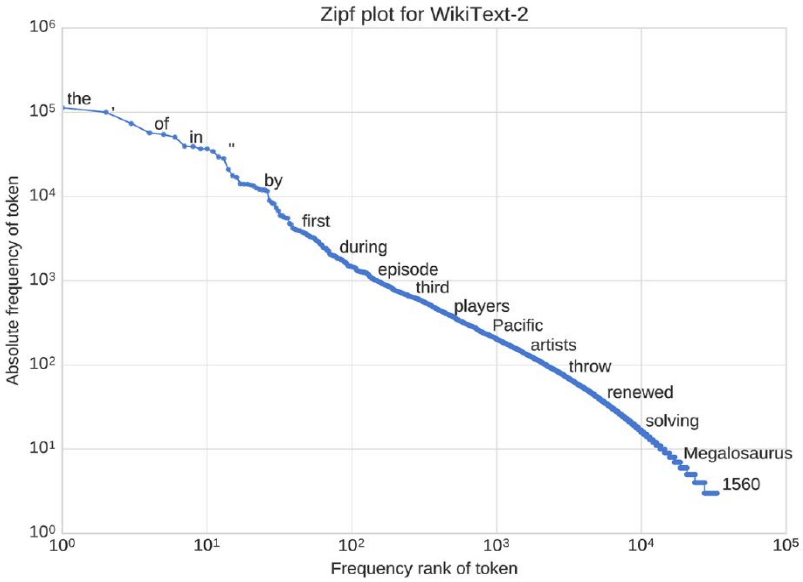
$$p_{\text{ptr}}(y_i|x_i) = \frac{1}{1-g}\, a[1:V],$$

https://arxiv.org/pdf/1609.07843.pdf

Datasets

	Penn Treebank			WikiText-2		
	Train	Valid	Test	Train	Valid	Test
Articles	-	-	-	600	60	60
Tokens	929,590	73,761	82,431	2,088,628	217,646	245,569
Vocab size	10,000			33,278		
OoV rate	4.8%			2.6%		





https://arxiv.org/pdf/1609.07843.pdf

Experiment Results

Model	Parameters	Validation	Test
Gal (2015) - Variational LSTM (medium, untied)	20M	81.9 ± 0.2	79.7 ± 0.1
Gal (2015) - Variational LSTM (medium, untied, MC)	20M	—	78.6 ± 0.1
Gal (2015) - Variational LSTM (large, untied)	66M	77.9 ± 0.3	75.2 ± 0.2
Gal (2015) - Variational LSTM (large, untied, MC)	66M	—	73.4 ± 0.0
Kim et al. (2016) - CharCNN	19M	—	78.9
Zilly et al. (2016) - Variational RHN	32M	72.8	71.3
Zoneout + Variational LSTM (medium)	20M	84.4	80.6
Pointer Sentinel-LSTM (medium)	21M	72.4	**70.9**

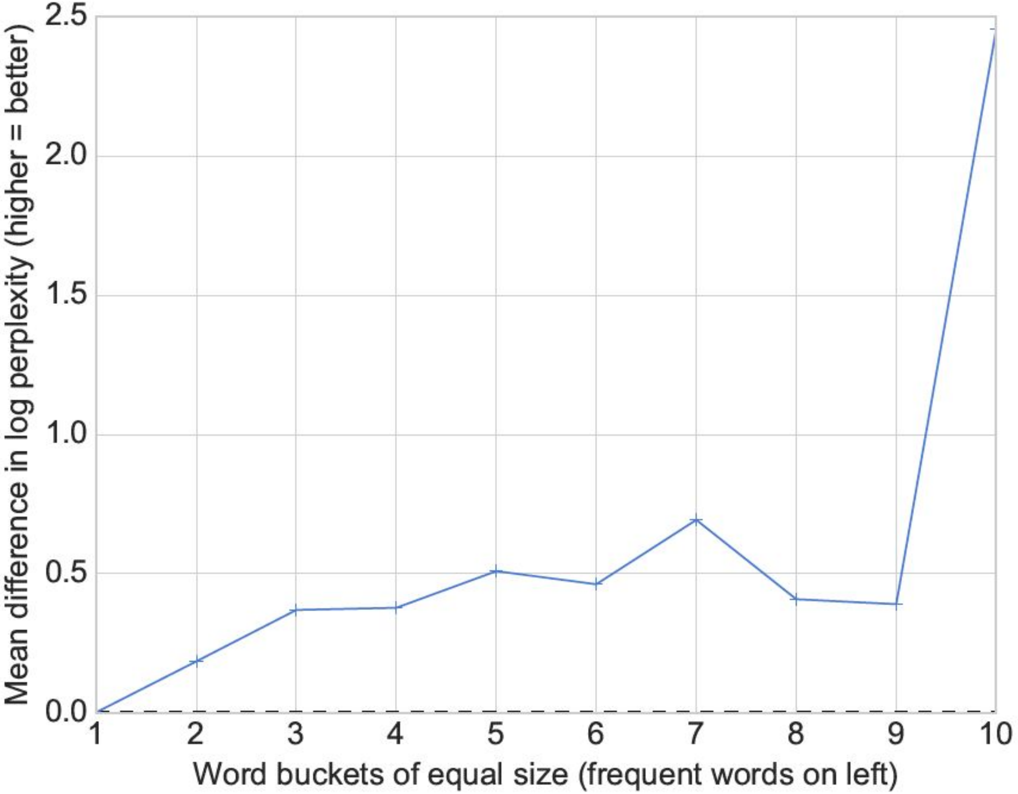
Perplexity on Penn Treebank

Model	Parameters	Validation	Test
Variational LSTM implementation from Gal (2015)	20M	101.7	96.3
Zoneout + Variational LSTM	20M	108.7	100.9
Pointer Sentinel-LSTM	21M	84.8	**80.8**

Perplexity on WikiText-2

https://arxiv.org/pdf/1609.07843.pdf

Impact on Rare Words



https://arxiv.org/pdf/1609.07843.pdf

From RNN to CNN

Limitations of current RNN LM that can be alleviated by CNN:

- They are blind to sub-word information. (Morphologically rich languages)
  - Solution: Character-Aware NLM (Kim et al., 2015)
- The computation of features or states for different parts of long sequences cannot occur in parallel
  - Solution: Quasi-RNN (Bradbury et al., 2017)

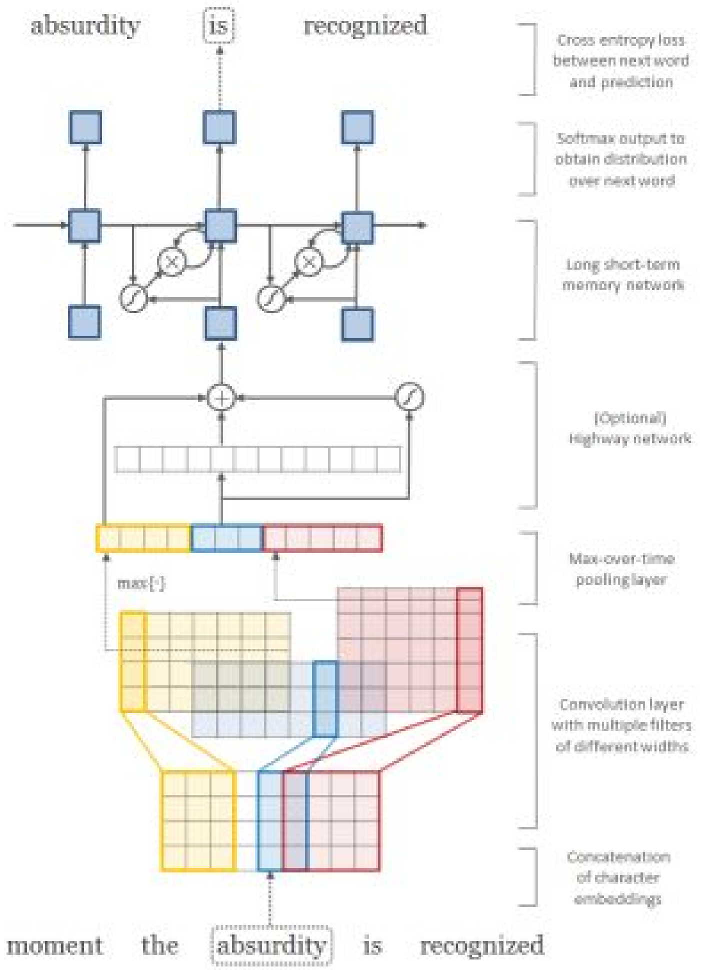
Character-Aware NLM

Highlights of the architecture:

- Instead of using word embeddings as input of RNN, (Kim et al., 2015) proposes to use the output of a character-level CNN as the input of RNN.
- The model has significantly fewer parameters as there is no word embedding involved.
- Highway network layer is added between CNN and RNN to boost performance.
    - Recap of highway network:

$$\mathbf{z} = \mathbf{t} \odot g(\mathbf{W}_H \mathbf{y} + \mathbf{b}_H) + (1 - \mathbf{t}) \odot \mathbf{y}$$

$$\mathbf{t} = \sigma(\mathbf{W}_T \mathbf{y} + \mathbf{b}_T)$$



https://arxiv.org/pdf/1508.06615.pdf

Experiments

	PPL	Size
LSTM-Word-Small	97.6	5 M
LSTM-CharCNN-Small	92.3	5 M
LSTM-Word-Large	85.4	20 M
LSTM-CharCNN-Large	78.9	19 M
Sum-Prod Net† (Cheng et al. 2014)	100.0	5 M
LSTM-Medium† (Zaremba et al. 2014)	82.7	20 M
LSTM-Large† (Zaremba et al. 2014)	78.4	52 M

Perplexity on Penn TreeBank (English)

		Cs	De	Es	Fr	Ru
B&B	KN-4	545	366	241	274	396
	MLBL	465	296	200	225	304
Small	Word	503	305	212	229	352
	Morph	414	278	197	216	290
	Char	397	250	174	203	284
Large	Word	493	286	200	222	357
	Morph	398	263	177	196	271
	Char	**375**	**238**	**163**	**184**	**269**

Perplexity on 2013 ACL Workshop on MT dataset

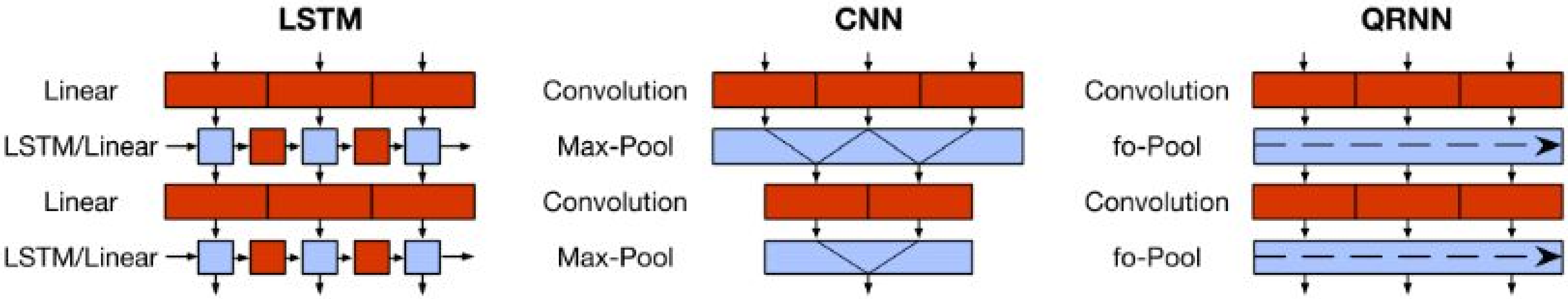
	Small	Large
No Highway Layers	100.3	84.6
One Highway Layer	92.3	79.7
Two Highway Layers	90.1	78.9
Multilayer Perceptron	111.2	92.6

Perplexity of models with different middle layers

https://arxiv.org/pdf/1508.06615.pdf

Quasi-RNN

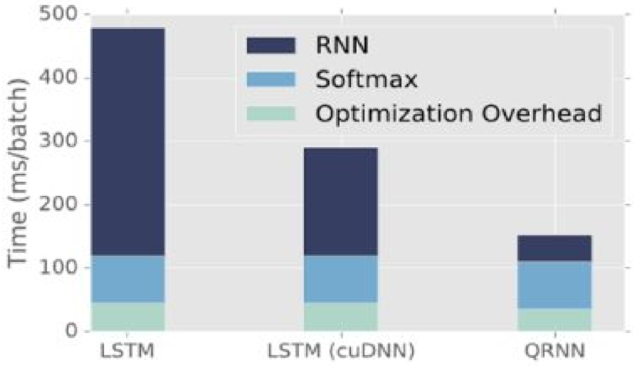
An approach to neural sequence modeling that alternates CNN, which apply in parallel across timesteps and a minimalist recurrent pooling function that applies in parallel across channels (Bradbury et al., 2017)



https://arxiv.org/pdf/1611.01576.pdf

Experiments

Model	Parameters	Validation	Test
LSTM (medium) (Zaremba et al., 2014)	20M	86.2	82.7
Variational LSTM (medium) (Gal & Ghahramani, 2016)	20M	81.9	79.7
LSTM with CharCNN embeddings (Kim et al., 2016)	19M	–	78.9
Zoneout + Variational LSTM (medium) (Merity et al., 2016)	20M	84.4	80.6
Our models			
LSTM (medium)	20M	85.7	82.0
QRNN (medium)	18M	82.9	79.9
QRNN + zoneout ($p = 0.1$) (medium)	18M	82.1	78.3



https://arxiv.org/pdf/1611.01576.pdf

Application of Deep Learning in NLP

- Question Answering
  - Dynamic Neural Networks
  - Improved Dynamic Neural Networks
  - Dynamic Co-attention Networks
- Coreference Resolution
  - Deep Reinforcement Learning for Mention- Ranking Coreference Models

Question Answering(QA) Example

I: Mary walked to the bathroom.
I: Sandra went to the garden.
I: Daniel went back to the garden.
I: Sandra took the milk there.
Q: Where is the milk?
A: garden
I: Everybody is happy.
Q: What's the sentiment?
A: positive

I: Jane has a baby in Dresden.
Q: What are the named entities?
A: Jane - person, Dresden - location
I: Jane has a baby in Dresden.
Q: What are the POS tags?
A: NNP VBZ DT NN IN NNP .
I: I think this model is incredible
Q: In French?
A: Je pense que ce modèle est incroyable.

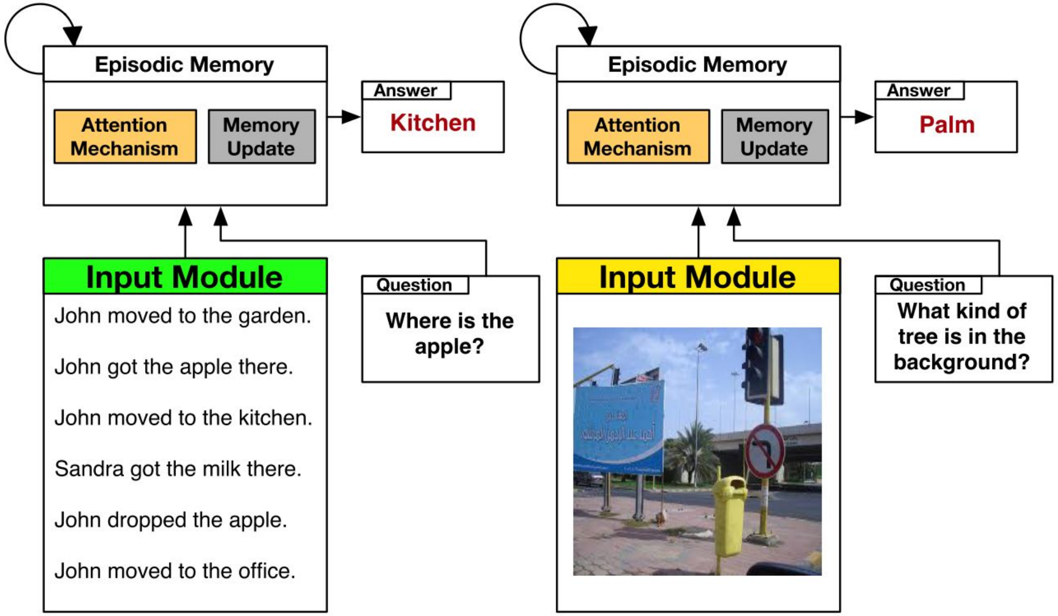
http://web.stanford.edu/class/cs224n/lectures/cs224n-2017-lecture16-DMN-QA.pdf

Dynamic Memory Network

- Dynamic Memory Network(Kumar et al., 2015)
  - Has both a memory component and an attention mechanism
- General Architecture for Question Answering（DMN+, Xiong et al., 2016）
  - Capable of tackling wide range of tasks and input formats
  - Can even been used for general NLP tasks (i.e. non QA) (PoS, NER, sentiment, translation, ...)
- Composed of different modules focusing on sub-tasks
  - Allows independent analysis and improvements on modules
    - Input representations
    - Memory components
    - etc

Dynamic Memory Network(DMN/DMN+)

- Modules
    - Input Module
    - Question Module
    - Episodic Memory Module
    - Answer Module

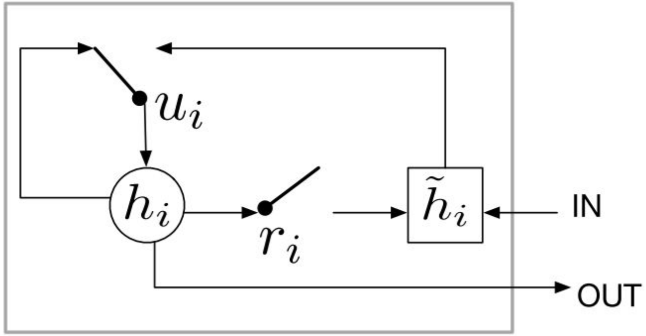


(a) Text Question-Answering (b) Visual Question-Answering

Caiming Xiong, Stephen Merity, and Richard Socher. [Dynamic Memory Networks for Visual and Textual Question Answering](). arXiv preprint arXiv:1603.01417, 2016.

DMN: Input Module

- Processes the input data (about which a question is being asked) into a set of vectors termed facts, represented as F = [f1, f2, ..., fN]
- Gated Recurrent Unit(GRU) networks are used



$$u_i = \sigma\left(W^{(u)}x_i + U^{(u)}h_{i-1} + b^{(u)}\right) \quad (1)$$

$$r_i = \sigma\left(W^{(r)}x_i + U^{(r)}h_{i-1} + b^{(r)}\right) \quad (2)$$

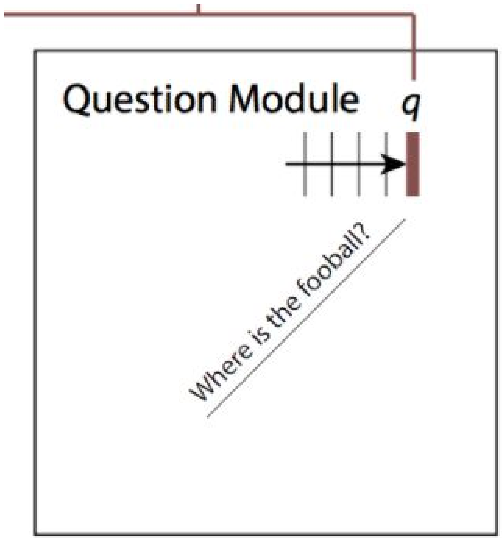
$$\tilde{h}_i = \tanh\left(Wx_i + r_i \circ Uh_{i-1} + b^{(h)}\right) \quad (3)$$

$$h_i = u_i \circ \tilde{h}_i + (1 - u_i) \circ h_{i-1} \quad (4)$$

Caiming Xiong, Stephen Merity, and Richard Socher. Dynamic Memory Networks for Visual and Textual Question Answering. arXiv preprint arXiv:1603.01417, 2016.

DMN: Modules

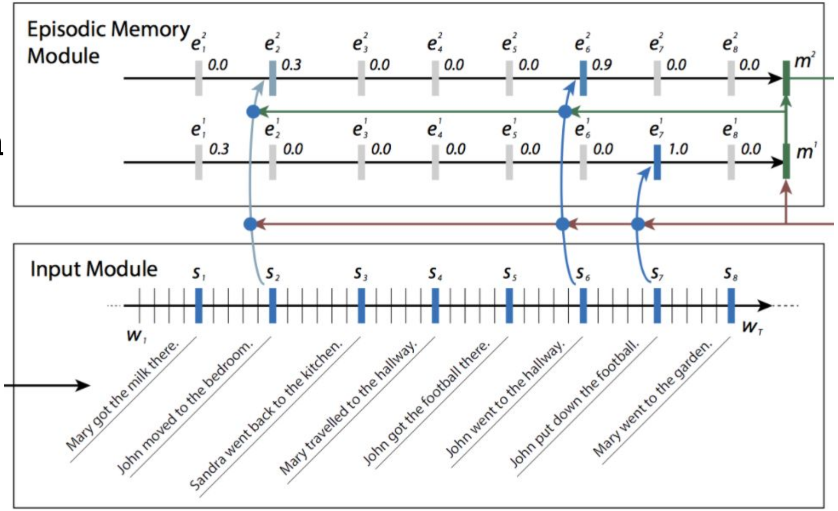
- Question Module: Maps question sentence to a vector representation (embedding) q
  - uses GRU
  - get last recurrent state



Kumar, A., Irsoy, O., Su, J., Bradbury, J., English, R., Pierce, B., … Socher, R. (2015). Ask Me Anything: Dynamic Memory Networks for Natural Language Processing. *arXiv*, 1–10.

DMN: Modules

- Episodic Memory Module: Retrieve information from facts F to answer question q.
  - May pass over input multiple times. Update memory vector m(t) after each pass.
  - The initial memory vector is set the question vector m(0)=q

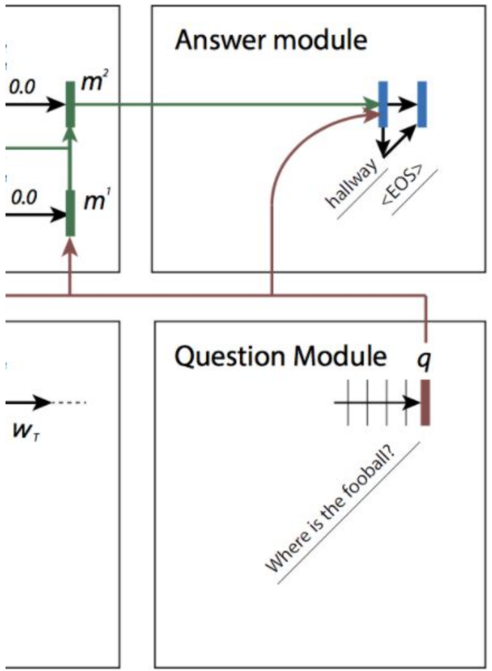


- Two components
  - Attention update mechanisms
    - Producing a contextual vector c(t)
    - Summary of relevant input
  - Memory update mechanisms
    - Generating the episode memory
    - Based upon c(t) and m(t-1)

Kumar, A., Irsoy, O., Su, J., Bradbury, J., English, R., Pierce, B., … Socher, R. (2015). Ask Me Anything: Dynamic Memory Networks for Natural Language Processing. *arXiv*, 1–10.

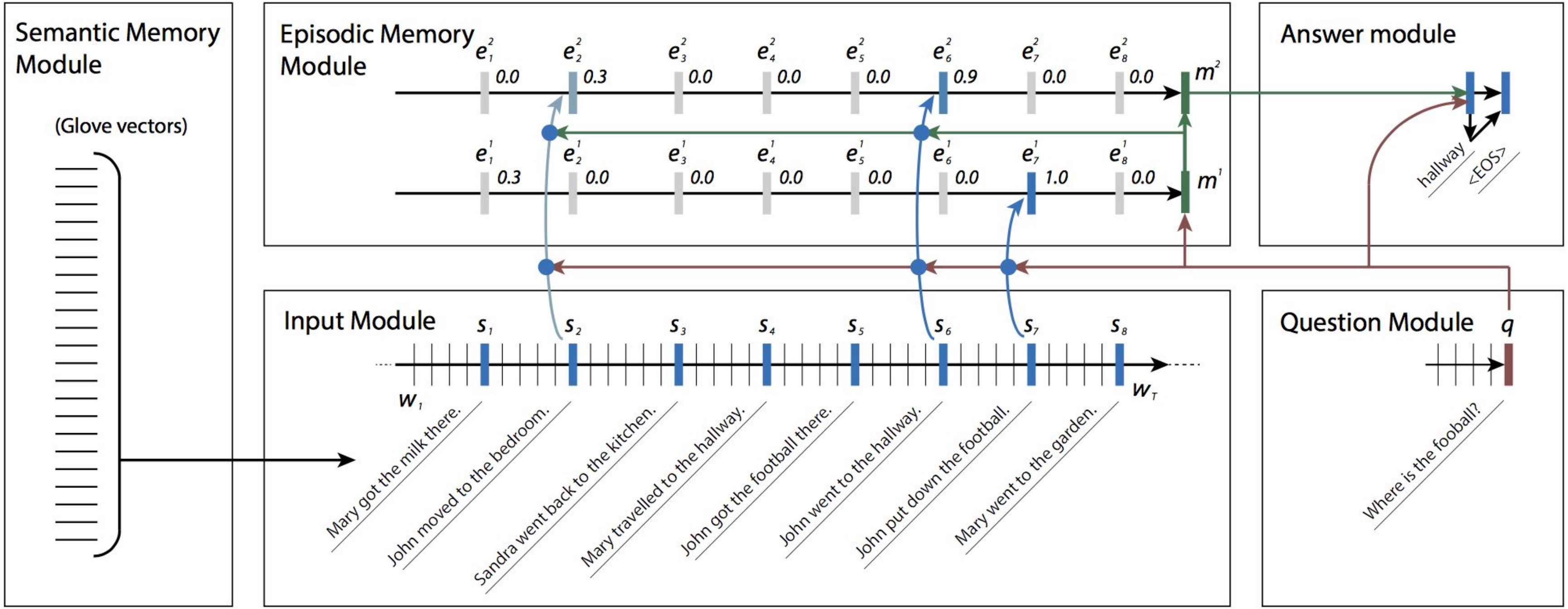
DMN: Modules

- Answer Module: receive both q and m(T) to generate the model's predicted answer
  - Simple one word answers: softmax output
  - Many words answers: RNN decoder to decode a = [q; m(T)]
- Training
  - Cross entropy error on the answers is used for training and backpropagate through the network



Kumar, A., Irsoy, O., Su, J., Bradbury, J., English, R., Pierce, B., … Socher, R. (2015). Ask Me Anything: Dynamic Memory Networks for Natural Language Processing. *arXiv*, 1–10.

DMN Recap



Kumar, A., Irsoy, O., Su, J., Bradbury, J., English, R., Pierce, B., … Socher, R. (2015). Ask Me Anything: Dynamic Memory Networks for Natural Language Processing. *arXiv*, 1–10.

DMN: Improvements

- While this worked well for bAbI-1k with supporting facts, it did not perform well on bAbI-10k without supporting facts
  - GRU only allows sentences to have context from sentences before them
  - Supporting sentences may be too far away from each other to allow for these distant sentences to interact through the word level GRU
- Improved Dynamic Memory Networks - DMN+(Xiong et al., 2016)
  - Input Fusion Layer
  - Updated episodic memory module

Model	ODMN
Input module	GRU
Attention	$\sum g_i f_i$
Mem update	GRU
Mem Weights	Tied
	bAbI Eng
QA2	36.0
QA3	42.2
QA5	0.1
QA6	35.7
QA7	8.0
QA8	1.6
QA9	3.3
QA10	0.6
QA14	3.6
QA16	55.1
QA17	39.6
QA18	9.3
QA20	1.9
Mean error	11.8

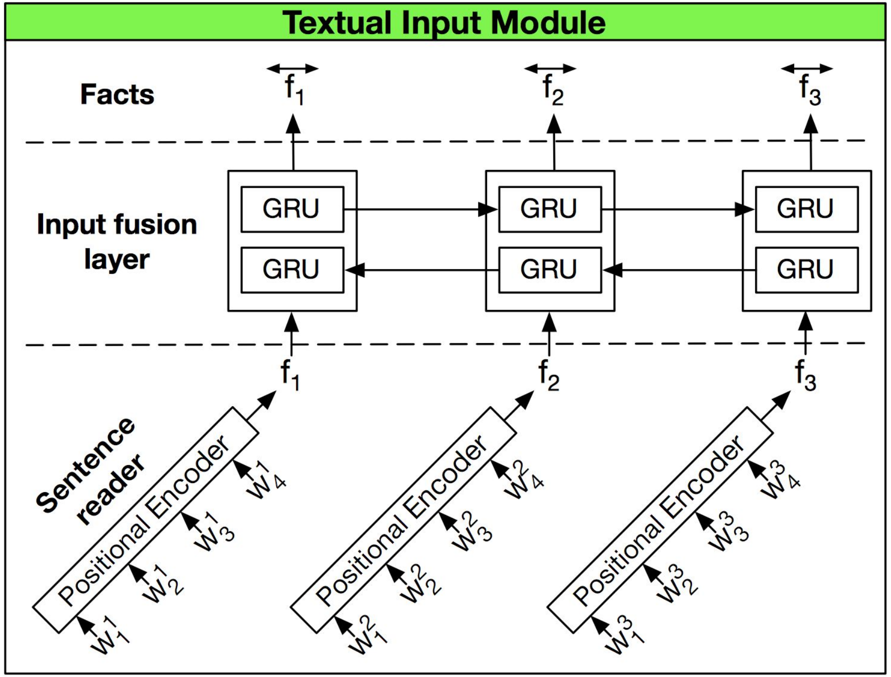
2: 2 supporting facts
3: 3 supporting facts
5: 3 argument relations
6: yes/no questions
7: counting
8: lists/sets
9: simple negation
11: basic coreference
14: time reasoning
16: basic induction
17: positional reasoning
18: size reasoning
19: path finding

Caiming Xiong, Stephen Merity, and Richard Socher. Dynamic Memory Networks for Visual and Textual Question Answering. arXiv preprint arXiv:1603.01417, 2016.

DMN+: Input Module

- Replace single GRU with two different components
  - Sentence reader: positional encoder is now used
  - Fusion layer: propagate information from future to generate facts

$$\overrightarrow{f_i} = GRU_{fwd}(f_i, \overrightarrow{f_{i-1}})$$
$$\overleftarrow{f_i} = GRU_{bwd}(f_i, \overleftarrow{f_{i+1}})$$
$$\overleftrightarrow{f_i} = \overleftarrow{f_i} + \overrightarrow{f_i}$$



Caiming Xiong, Stephen Merity, and Richard Socher. [Dynamic Memory Networks for Visual and Textual Question Answering](). arXiv preprint arXiv:1603.01417, 2016.

DMN+: Episodic Memory Module

- Episodic Memory Module: Retrieve information from input facts F by focusing attention on a subset of these facts
- Compute scalar attention gate value g^t_i with each fact f(i) during pass t.
    - Computation allows interactions between fact, question and episode memory state m(t-1)
- Gates g are activated if the sentence relevant to the question or memory

$$z_i^t = [\overleftrightarrow{f_i} \circ q; \overleftrightarrow{f_i} \circ m^{t-1}; |\overleftrightarrow{f_i} - q|; |\overleftrightarrow{f_i} - m^{t-1}|]$$

$$Z_i^t = W^{(2)} \tanh\left(W^{(1)} z_i^t + b^{(1)}\right) + b^{(2)}$$

$$g_i^t = \frac{\exp(Z_i^t)}{\sum_{k=1}^{M_i} \exp(Z_k^t)} \qquad ($$

Caiming Xiong, Stephen Merity, and Richard Socher. Dynamic Memory Networks for Visual and Textual Question Answering. arXiv preprint arXiv:1603.01417, 2016.

DMN+: Episodic Memory

- Now with each gating per fact g(i,t), we can get the contextual vector c(t) to update the memory m(t)
- Two options
  - Soft attention: apply the softmax weights directly over the facts
    - Advantages
      - Easy to compute
      - If the softmax activation is spiky, it can approximate a hard attention
    - Disadvantage
      - summation loses positional and ordering information

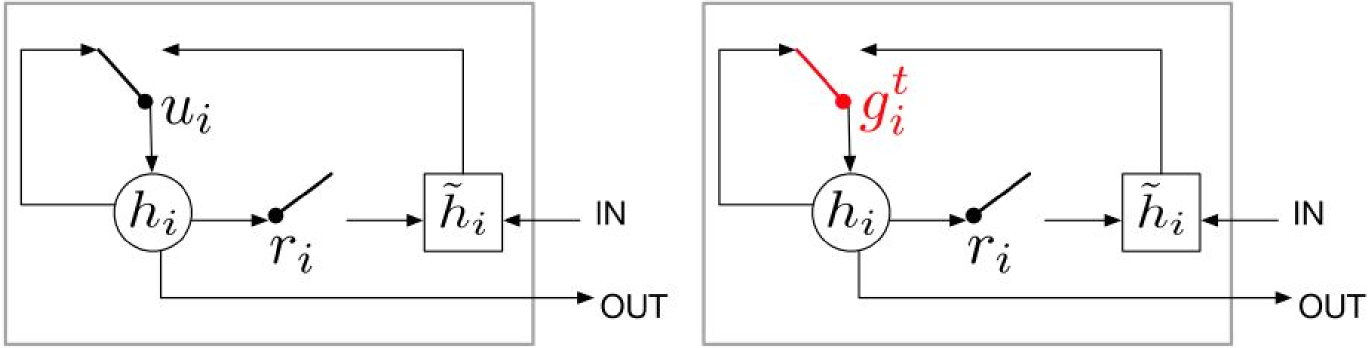
$$\sum_{i=1}^{N} g_i^t \overleftrightarrow{f}_i$$

  - Attention based GRU
    - More sensitive to both the position and ordering of the input facts F

DMN+: Episodic Memory

- Attention based GRU: attention should be sensitive to position and ordering of input facts F
  - Update gates decides how much of each dimension of hidden states to retain and how much should be updated at every timestep
  - Replace update gate for the attention gate

$$h_i = u_i \circ \tilde{h}_i + (1 - u_i) \circ h_{i-1} \qquad h_i = g_i^t \circ \tilde{h}_i + (1 - g_i^t) \circ h_{i-1}$$



Caiming Xiong, Stephen Merity, and Richard Socher. Dynamic Memory Networks for Visual and Textual Question Answering. arXiv preprint arXiv:1603.01417, 2016.

DMN+: Experiments

- Dataset
  - Facebook bAbI-10k

- ODMN: original DMN
- DMN2: Input Fusion Layer
- DMN3: Attention based GRU
- DMN+: ReLU activation to compute memory update

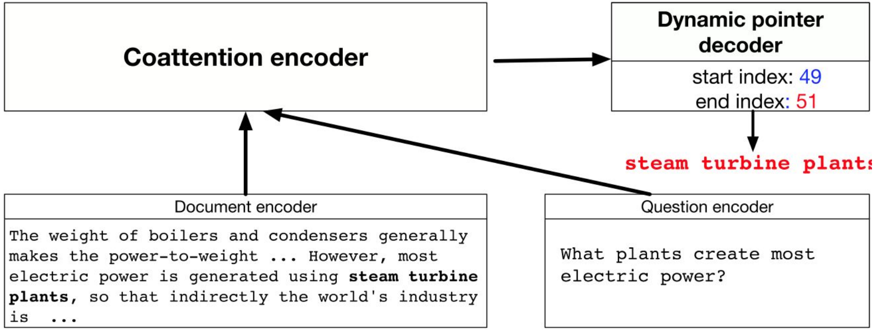
Model	ODMN	DMN2	DMN3	DMN+
Input module	GRU	Fusion	Fusion	Fusion
Attention	$\sum g_i f_i$	$\sum g_i f_i$	AttnGRU	AttnGRU
Mem update	GRU	GRU	GRU	ReLU
Mem Weights	Tied	Tied	Tied	Untied
bAbI English 10k dataset				
QA2	36.0	0.1	0.7	0.3
QA3	42.2	19.0	9.2	1.1
QA5	0.1	0.5	0.8	0.5
QA6	35.7	0.0	0.6	0.0
QA7	8.0	2.5	1.6	2.4
QA8	1.6	0.1	0.2	0.0
QA9	3.3	0.0	0.0	0.0
QA10	0.6	0.0	0.2	0.0
QA14	3.6	0.7	0.0	0.2
QA16	55.1	45.7	47.9	45.3
QA17	39.6	5.9	5.0	4.2
QA18	9.3	3.8	0.1	2.1
QA20	1.9	0.0	0.0	0.0
Mean error	11.8	3.9	3.3	2.8
DAQUAR-ALL visual dataset				
Accuracy	27.54	28.43	28.62	28.79

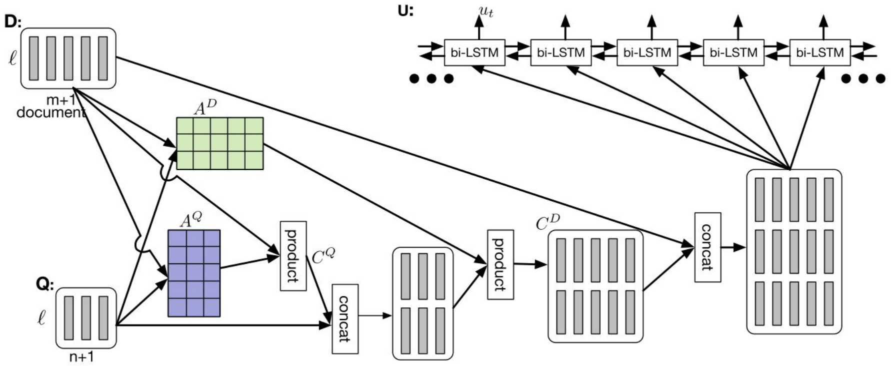
2: 2 supporting facts
3: 3 supporting facts
5: 3 argument relations
6: yes/no questions
7: counting
8: lists/sets
9: simple negation
11: basic coreference
14: time reasoning
16: basic induction
17: positional reasoning
18: size reasoning
19: path finding

Caiming Xiong, Stephen Merity, and Richard Socher. Dynamic Memory Networks for Visual and Textual Question Answering. arXiv preprint arXiv:1603.01417, 2016.

Dynamic Coattention Networks(Xiong et al., 2017)

- Coattentive encoder that captures the interactions between the question and the document
    - Attend to the question and document simultaneously
- a dynamic pointing decoder that alternates between estimating the start and end of the answer span





Caiming Xiong , Victor Zhong, and Richard Socher. [Dynamic Coattention Networks for Question Answering](). ICLR 2017.

QA Demo

Bi-directional Attention Flow Demo

for Stanford Question Answering Dataset (SQuAD)

Direction : Select a paragraph and write your own question. The answer is always a subphrase of the paragraph - remember it when you ask a question!

Select Paragraph

[36] Yuan dynasty

Paragraph

The Yuan dynasty (Chinese: 元朝; pinyin: Yuán Cháo), officially the Great Yuan (Chinese: 大元; pinyin: Dà Yuán; Mongolian: Yehe Yuan Ulus[a]), was the empire or ruling dynasty of China established by Kublai Khan, leader of the Mongolian Borjigin clan. Although the Mongols had ruled territories including today's North China for decades, it was not until 1271 that Kublai Khan officially proclaimed the dynasty in the traditional Chinese style. His realm was, by this point, isolated from the other khanates and controlled most of present-day China and its surrounding areas, including modern Mongolia and Korea. It was the first foreign dynasty to rule all of China and lasted until 1368, after which its Genghisid rulers returned to their Mongolian homeland and continued to rule the Northern Yuan dynasty. Some of the

Question

When was Yuan Dynasty established?

new question!

Answer

1271

Reference : Minjoon Seo, Aniruddha Kembhavi, Ali Farhadi, Hannaneh Hajishirzi. "Bidirectional Attention Flow for Machine Comprehension" [link]

Demo by : Sewon Min

https://arxiv.org/abs/1611.01603

Coreference Resolution

- What is Coreference Resolution ?
  - Identify all noun phrases(mentions) that refer

Barack Obama nominated Hillary Rodham Clinton as his secretary of state on Monday. He chose her because she had foreign affairs experience as a former First Lady.

- Applications
  - Full text understanding
  - Machine translation
  - Text summarization
  - information extraction and question answering

Coreference Resolution

- What is Coreference Resolution ?
  - Identify all noun phrases(mentions) that refer
  - Coreference resolution is a document-level structured prediction task

Barack Obama nominated Hillary Rodham Clinton as his secretary of state on Monday. He chose her because she had foreign affairs experience as a former First Lady.

- Applications
  - Full text understanding
  - Machine translation
  - Text summarization
  - information extraction and question answering

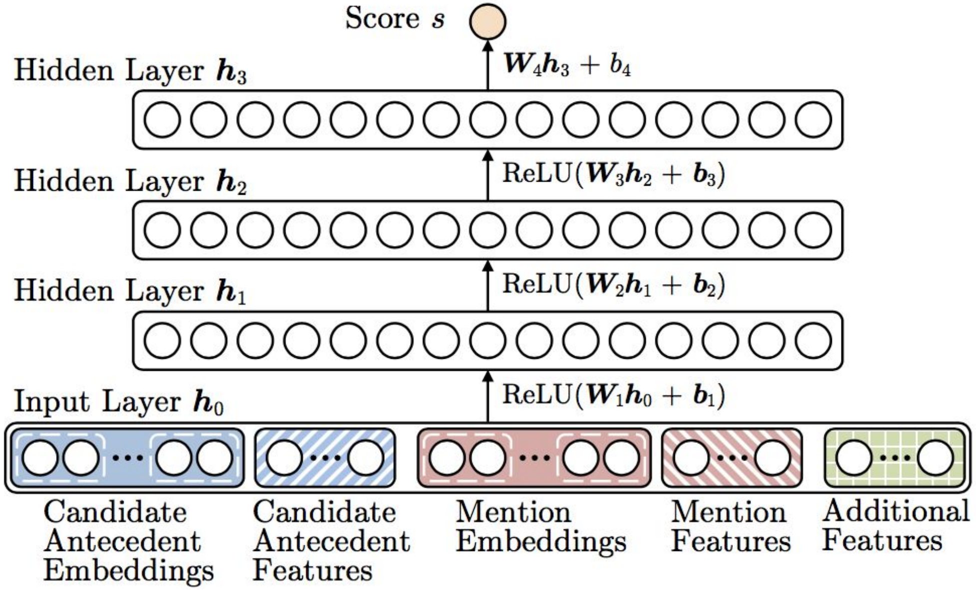
Coreference Models

- Mention Pair models
  - Treat coreference chains as a collection of pairwise links
  - Make independent pairwise decisions
  - Reconcile them in some deterministic way (e.g. transitivity)
- Mention-Ranking Models
  - Dominant approach to coreference resolution in recent years
  - Assign each mention its highest scoring candidate antecedent according to the model
  - Infer global structure by making a sequence of local decisions
- Entity-Mention models
  - A cleaner, but less studied approach
  - Explicitly cluster mentions of the same discourse entity

http://web.stanford.edu/class/cs224n/lectures/cs224n-2017-lecture15.pdf

Neural Mention-Pair Model

- Standard feed-forward neural network
  - From (Clark and Manning, 2016); similar to Wiseman et al. (2015)
  - Input layer: word embeddings and a few categorical features



http://cs.stanford.edu/people/kevclark//resources/clark-manning-emnlp2016-deep.pdf

Neural Mention-Pair Model

- Experiment
  - Dataset: English and Chinese Portions of the CoNLL 2012 Shared Task dataset

Model	English	Chinese
Chen & Ng (2012) [CoNLL 2012 Chinese winner]	54.52	**57.63**
Fernandes (2012) [CoNLL 2012 English winner]	**60.65**	51.46
Björkelund & Kuhn. (2014) Best previous Chinese system]	61.63	**60.06**
Wiseman et al. (2016) [Best previous English system]	**64.21**	—
Clark & Manning (ACL 2016)	65.29	63.66

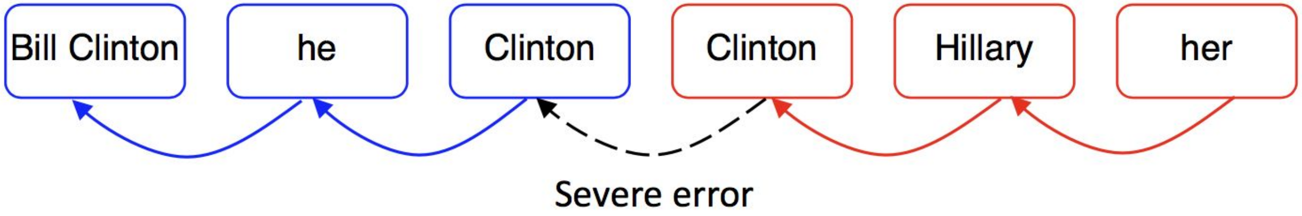
Example Wins

Anaphor	Antecedent
the country's leftist rebels	the guerillas
the company	the New York firm
216 sailors from the ``USS cole''	the crew
the gun	the rifle

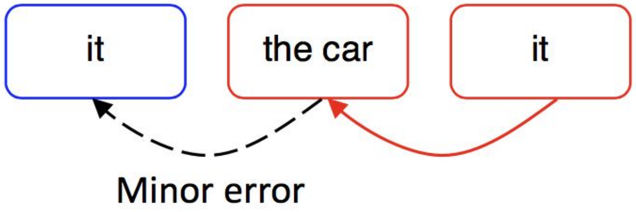
http://cs.stanford.edu/people/kevclark//resources/clark-manning-emnlp2016-deep.pdf

Neural Mention-Pair Model

- Next Challenge: Some Local Decisions Matter More than others



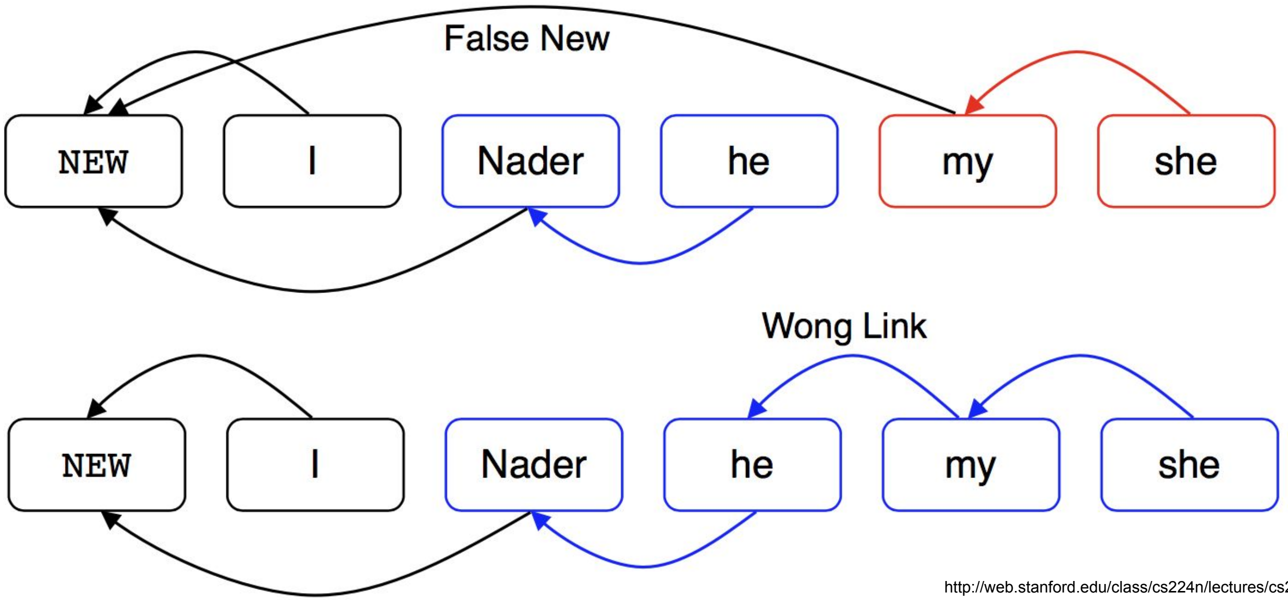
*"**it** was raining, but **the car** stayed dry because **it** was under cover"*



http://web.stanford.edu/class/cs224n/lectures/cs224n-2017-lecture15.pdf

Prior work

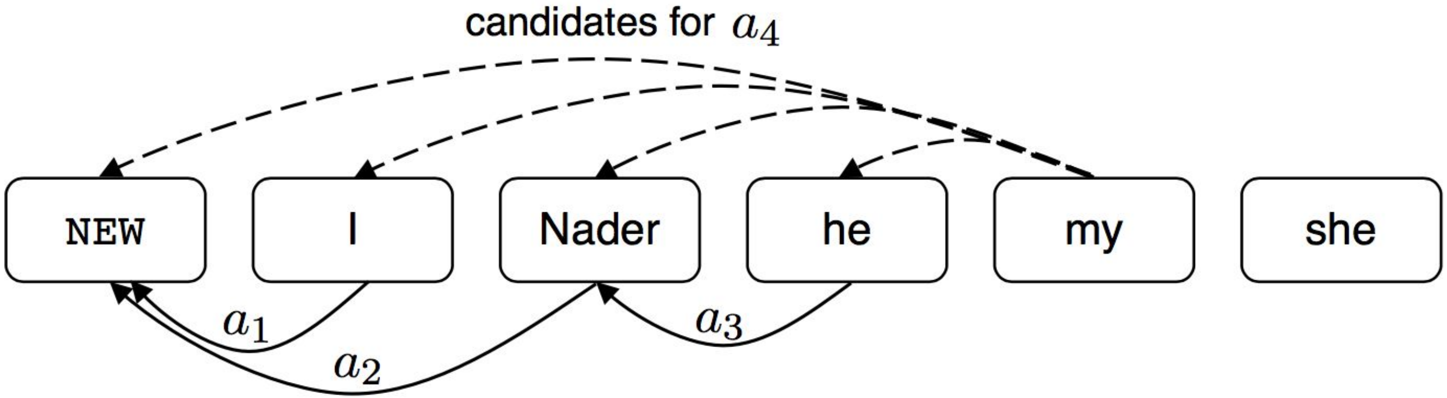
- Heuristically defined the importance of a coreference decision
  - Requires careful tuning with hyperparameters - Grid Search



http://web.stanford.edu/class/cs224n/lectures/cs224n-2017-lecture15.pdf

Coreference Resolution - RL

- Reinforcement Learning
- Clark & Manning(EMNLP 2016): use RL to learn which local decisions lead to a good clustering
  - No hyperparameter search
  - Small boost in accuracy



http://web.stanford.edu/class/cs224n/lectures/cs224n-2017-lecture15.pdf

Coreference Resolution - RL

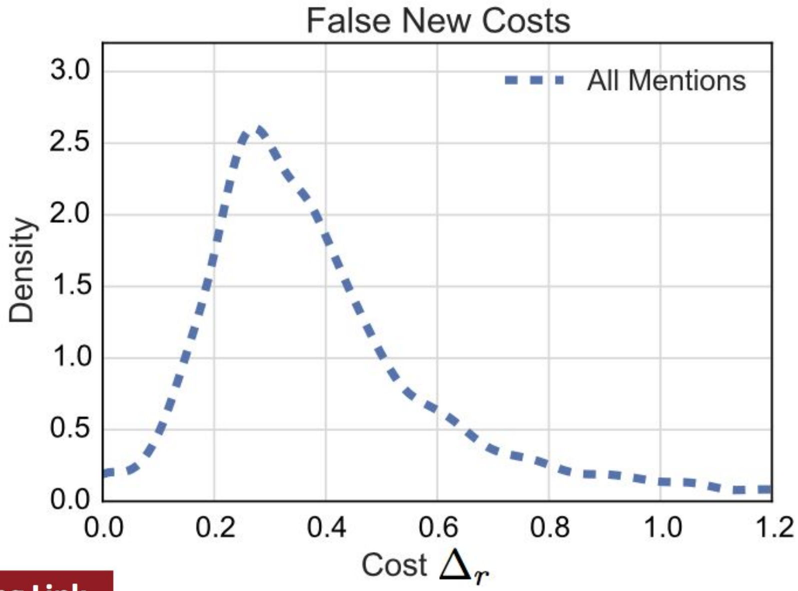
- Training
  - After completing a sequence of actions, the model receives a reward
  - Examining Reward-Based Costs
- Experiment
  - Dataset: English and Chinese Portions of the CoNLL 2012 Shared Task dataset

Model	English	Chinese
Heuristic Loss	65.36	63.54
REINFORCE	65.41	63.64
Reward Rescaling	**65.73**	**63.88**

http://cs.stanford.edu/people/kevclark//resources/clark-manning-emnlp2016-deep.pdf

Coreference Resolution - RL

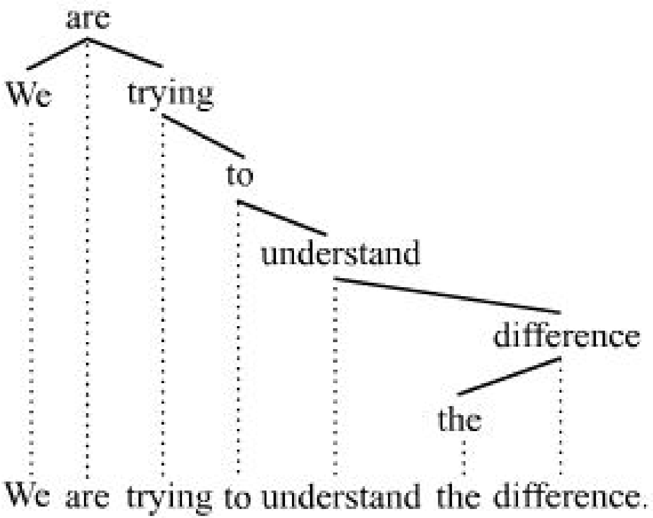
- Error Breakdown
  - Reinforcement learning model actually makes more errors!
  - However, the errors are less severe
- Reward-Based Costs
  - High variance in costs for a given error type



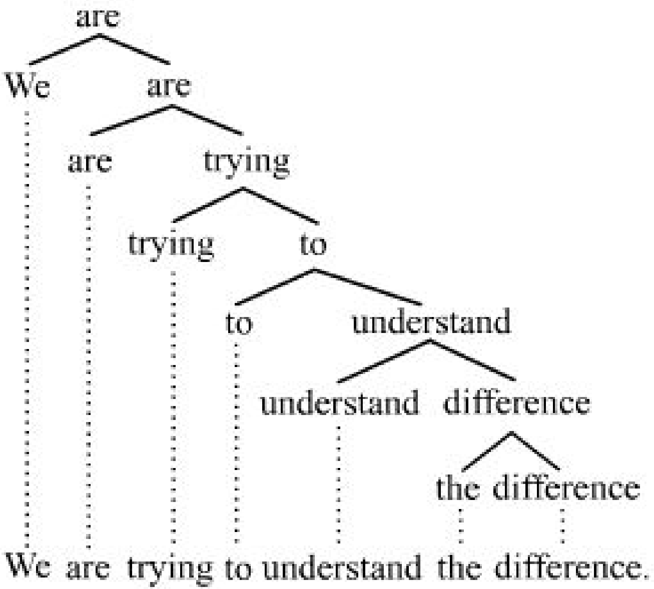
Model	# False Anaphoric	# False New	# Wrong Link
Heuristic Loss	1956	1719	1258
Reward Rescaling	1994	1725	1247

http://cs.stanford.edu/people/kevclark//resources/clark-manning-emnlp2016-deep.pdf

Syntactic Parsing (Dependency & Constituency)



Dependency



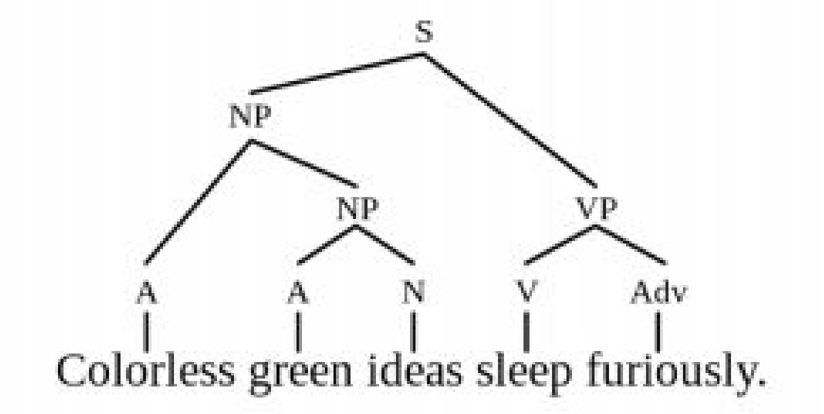
Constituency (BPS)

Parsing as Tools in NLP

- Resolve ambiguities in language:
  - E.g. “I saw a girl with a telescope.”
- Provide more information as additional features in NLP tasks:
  - Entity Recognition
  - Relation Extraction
  - Word embedding learning
  - ...

Constituency Parsing

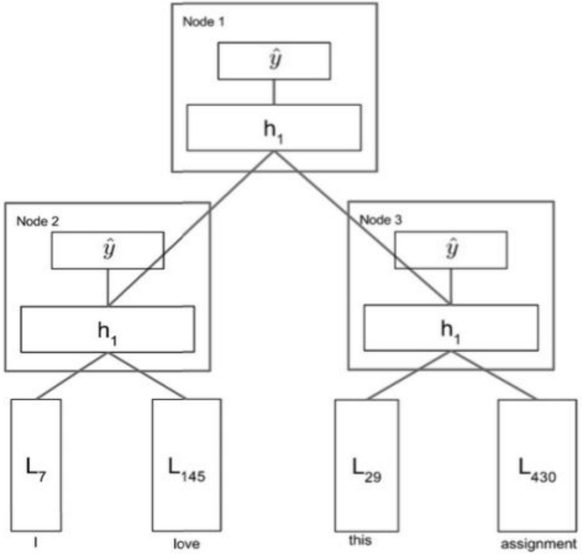
Constituency Parsing (phrase structure parsing) is a way to break a piece of text (e.g. one sentence) into sub-phrases. One goal is to identify the constituents which would be useful when extracting information from text.



http://web.stanford.edu/class/cs224n/lecture_notes/cs224n-2017-notes7.pdf

Recursive Neural Networks

Recursive Neural Networks (Tree RNNs) are perfect for settings that have nested hierarchy and an intrinsic recursive structure.



$$h^{(1)} = \tanh\left(W^{(1)} \begin{bmatrix} h^{(1)}_{Left} \\ h^{(1)}_{Right} \end{bmatrix} + b^{(1)}\right)$$

http://web.stanford.edu/class/cs224n/lecture_notes/cs224n-2017-notes7.pdf

Tree RNNs for Structure Prediction

Structured margin loss :

ground truth tree

$$\Delta(x, l, \hat{y}) = \kappa \sum_{d \in N(\hat{y})} \mathbf{1}\{subTree(d) \notin Y(x, l)\}$$

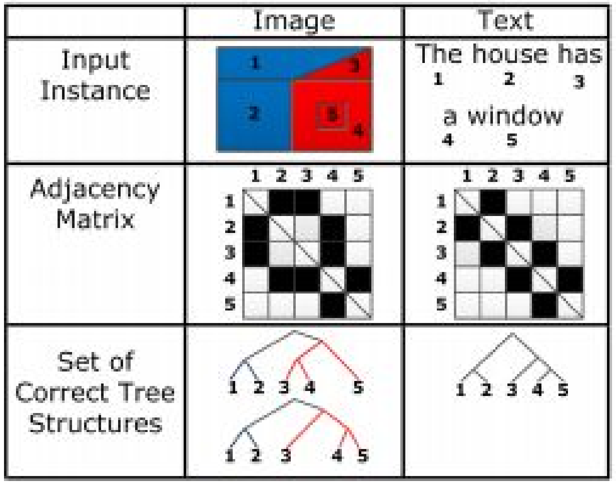
the set of non-terminal nodes

Regularized risk function to be minimized:

$$J(\theta) = \frac{1}{N} \sum_{i=1}^{N} r_i(\theta) + \frac{\lambda}{2} ||\theta||^2$$

$$\begin{aligned} r_i(\theta) &= \max_{\hat{y} \in T(x_i)} \left(s(\mathrm{RNN}(\theta, x_i, \hat{y})) + \Delta(x_i, l_i, \hat{y}) \right) \\ &- \max_{y_i \in Y(x_i, l_i)} \left(s(\mathrm{RNN}(\theta, x_i, y_i)) \right) \end{aligned}$$

all possible trees that can be constructed from an input x



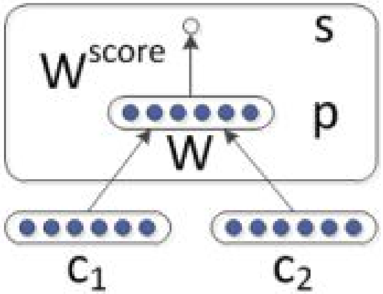
Training Inputs

http://ai.stanford.edu/~ang/papers/icml11-ParsingWithRecursiveNeuralNetworks.pdf

Tree RNNs for Structure Prediction

Greedy Structure Predicting RNNs (Socher et al., 2011)

- After computing the scores for all pairs of neighboring segments, the algorithm selects the pair which received the highest score.
- The process repeats (treating the new pi,j just like any other segment) until all pairs are merged and only one parent activation is left in the set C.



One recursive neural network which is replicated for each pair of possible input vectors

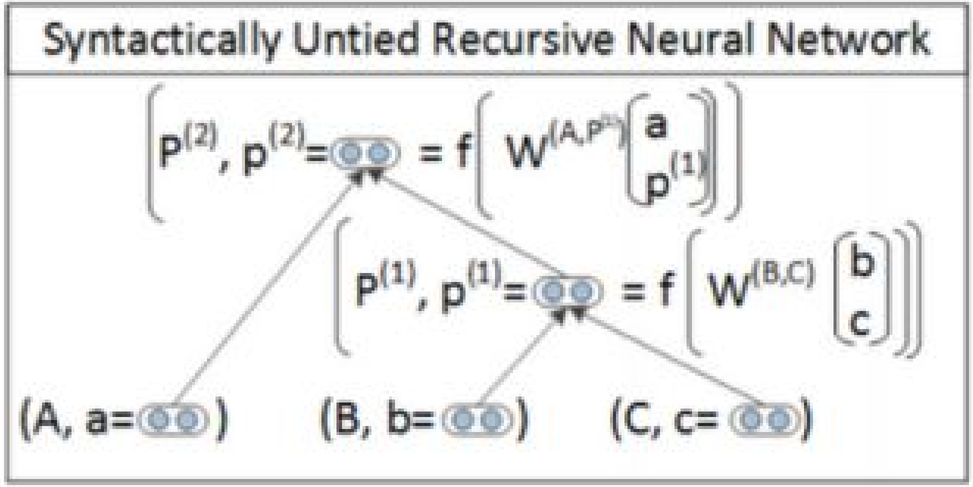
$$s = W^{score} p$$
$$p = f(W[c_1; c_2] + b)$$

$$s(\mathrm{RNN}(\theta, x_i, \hat{y})) = \sum_{d \in N(\hat{y})} s_d$$

http://ai.stanford.edu/~ang/papers/icml11-ParsingWithRecursiveNeuralNetworks.pdf

Syntactically Untied Tree RNN (SU-RNN)

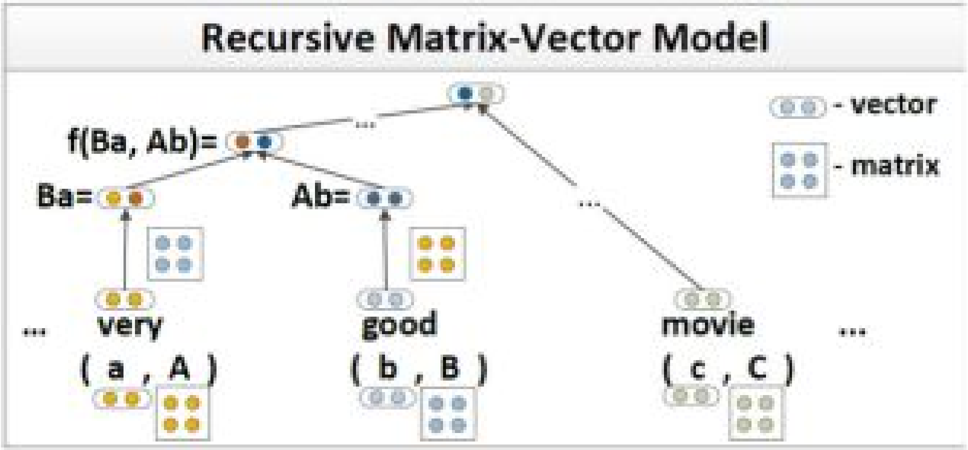
Using different W's for different categories of inputs: "syntactically untie" the weights of these different tasks. (Socher et al., 2013a)



http://web.stanford.edu/class/cs224n/lecture_notes/cs224n-2017-notes7.pdf

Matrix-Vector Tree RNN (MV-RNN)

We now augment our word representation, to not only include a word vector, but also a word matrix (more expressive)! (Socher et al., 2012)



http://web.stanford.edu/class/cs224n/lecture_notes/cs224n-2017-notes7.pdf

Tree RNN for Classification

We can leverage the vector representation of each node by adding to each RNN parent node a simple softmax layer to predict class labels, such as visual or syntactic categories.

$$label_p = softmax(W^{label} p)$$

http://ai.stanford.edu/~ang/papers/icml11-ParsingWithRecursiveNeuralNetworks.pdf

Sentiment Classification with Tree RNN

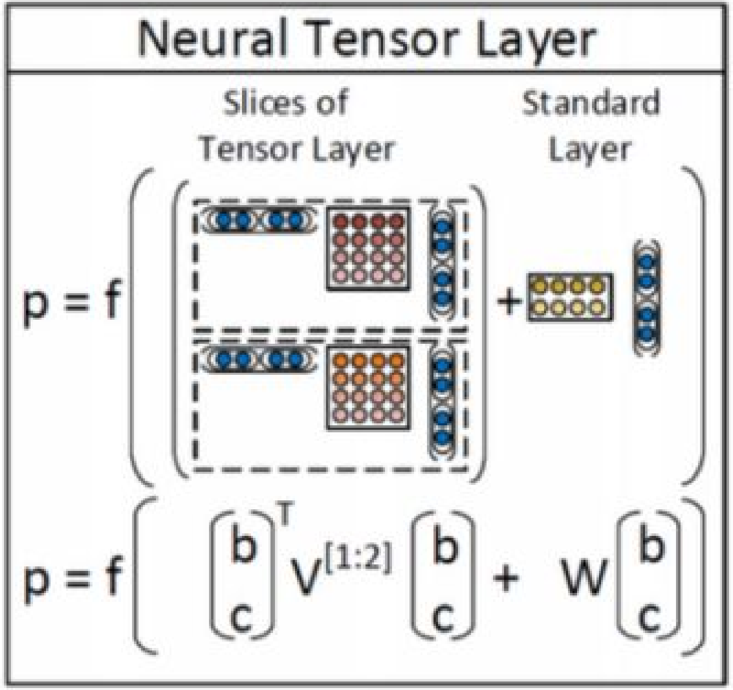
(Socher et al., 2013b) applies Recursive Neural Tensor Network (RNTN) for the task of sentiment analysis (5 sentiment classes)

$h^{(1)} = tanh(x^T V x + W x)$ $\quad V$ is a 3rd order tensor in $\in \mathbb{R}^{2d \times 2d \times d}$

$x^T V[i] x \; \forall i \in [1, 2, \ldots d]$ slices of the tensor outputting a vector $\in \mathbb{R}^d$

http://web.stanford.edu/class/cs224n/lecture_notes/cs224n-2017-notes7.pdf

Sentiment Classification with Tree RNN



One slice of a RNTN. There would be d of these slices.

(Socher et al., 2013b)

https://nlp.stanford.edu/~socherr/EMNLP2013_RNTN.pdf

Conclusion

- Summary
  - Introduction to Natural Language Processing
  - Word Representation
  - Language Model
  - Question Answering
  - Coreference Resolution
  - Dependency Parsing
  - Constituency Parsing

- Limitation & Challenges
  - limited in their ability to "reason": e.g. A dog is chasing the boy.
  - Need too much data (in a supervised fashion)
  - And more

References

- Word Representations:
  - Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient Estimation of Word Representations in Vector Space. ICLR, 2013.
  - Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. GloVe: Global Vectors for Word Representation. EMNLP, 2014.
- Language Models:
  - Stephen Merity, Caiming Xiong, James Bradbury, and Richard Socher. Pointer sentinel mixture models. arXiv preprint arXiv:1609.07843, 2016.
  - James Bradbury, Stephen Merity, Caiming Xiong, and Richard Socher. Quasi-recurrent neural networks. arXiv preprint arXiv:1611.01576, 2016.
  - Yoon Kim, Yacine Jernite, David Sontag, and Alexander Rush. Character-Aware Neural Language Models. AAAI, 2016.

References

- Question Answering:
  - Caiming Xiong, Stephen Merity, and Richard Socher. Dynamic Memory Networks for Visual and Textual Question Answering. arXiv preprint arXiv:1603.01417, 2016.
  - Caiming Xiong , Victor Zhong, and Richard Socher. Dynamic Coattention Networks for Question Answering. ICLR 2017.
- Coreference Resolution:
  - Sam Wiseman and Alexander M. Rush and Stuart M. Shieber. Learning Global Features for Coreference Resolution, NAACL 2016.
  - Kevin Clark and Christopher D. Manning. Improving Coreference Resolution by Learning Entity-Level Distributed Representations. ACL, 2016.
  - Kevin Clark and Christopher D. Manning. Deep Reinforcement Learning for Mention-Ranking Coreference Models, EMNLP 2016.

References

- Constituency Parsing:
    - Richard Socher, John Bauer, Christopher D. Manning and Andrew Y. Ng. Parsing with Compositional Vector Grammars. ACL, 2013.
    - Richard Socher, Brody Huval, Christopher D Manning, and Andrew Y Ng. Semantic compositionality through recursive matrix-vector spaces. EMNLP-CoNLL, 2012.
    - Richard Socher, Alex Perelygin, Jean Y. Wu, Jason Chuang, Christopher D. Manning, Andrew Y. Ng and Christopher Potts. Recursive Deep Models for Semantic Compositionality Over a Sentiment Treebank. EMNLP, 2013.
    - Richard Socher, Cliff Chiung-Yu Lin, Andrew Y. Ng and Christopher D. Manning. Parsing Natural Scenes and Natural Language with Recursive Neural Networks. ICML, 2011.
- Dependency Parsing:
    - Daniel Andor, Chris Alberti, David Weiss, Aliaksei Severyn, Alessandro Presta, Kuzman Ganchev, Slav Petrov, Michael Collins. Globally normalized transition-based neural networks. ACL, 2016.

Backup Slides

DMN+: Experiments

- Comparison with another state o
  architectures
  - End to end memory network
  - Neural reasoner framework

Task	DMN+	E2E	NR
2: 2 supporting facts	0.3	0.3	-
3: 3 supporting facts	1.1	2.1	-
5: 3 argument relations	0.5	0.8	-
6: yes/no questions	0.0	0.1	-
7: counting	2.4	2.0	-
8: lists/sets	0.0	0.9	-
9: simple negation	0.0	0.3	-
11: basic coreference	0.0	0.1	-
14: time reasoning	0.2	0.1	-
16: basic induction	45.3	51.8	-
17: positional reasoning	4.2	18.6	0.9
18: size reasoning	2.1	5.3	-
19: path finding	0.0	2.3	1.6
Mean error (%)	2.8	4.2	-
Failed tasks (err >5%)	1	3	-

http://www.jmlr.org/proceedings/papers/v48/xiong16.pdf

DMN+: Episodic Memory

- After each pass through the attention mechanism, we update m(t)
  - $m^t = GRU(c^t, m^{t-1})$
- How to obtain context vector c(t) with attention GRU
  - Last recurrent state
- Changing to ReLU improves accuracy by another 0.5%
  - $m^t = ReLU\left(W^t[m^{t-1}; c^t; q] + b\right)$