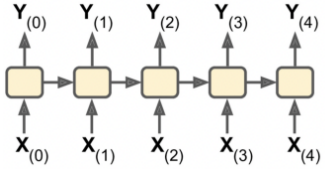
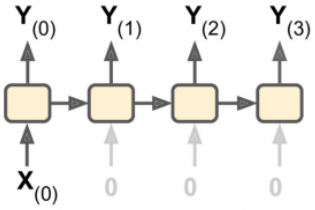
CAS CS 505
Lec 12

BRNNs; Applications of (B)RNNs: POS Tagging, Named Entity Recognition; Embeddings Revisited

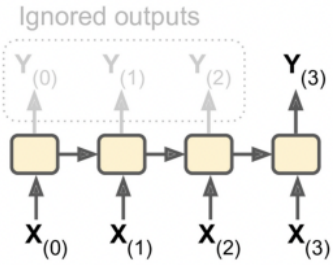
1. Review: RNN Architectures
    a. Sequence-to-Sequence (Sequence of vectors to Sequence of Vectors)



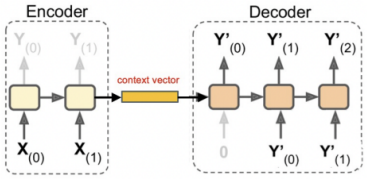
    b. Vector-to-Sequence



    c. Sequence-to-Vector (classification)

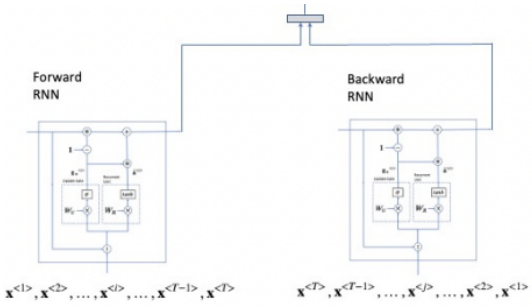


    d. Encoder-Decoder Combination



        i. Encoder → Sequence-to-Vector model and passes the context vector
        ii. Diagrams are unrolled over time
        iii. Context is decoded into sentences → generates sentences
        iv. Decoder → Vector-to-Sequence model
2. Bidirectional Recurrent Neural Networks (BRNNs)

a. Two RNNs may be combined to look at the sequence in the forward and backward direction at the same time!

b. 

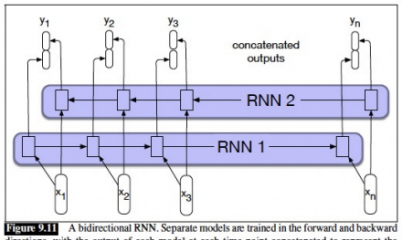
c. 

Figure 9.11 A bidirectional RNN. Separate models are trained in the forward and backward directions, with the output of each model at each time point concatenated to represent the bidirectional state at that time point.

d. Bidirectional RNNs make TWO separate passes through the input sequence, storing the activations from the first pass for the second pass. (two layers that look at the sequence from both directions)

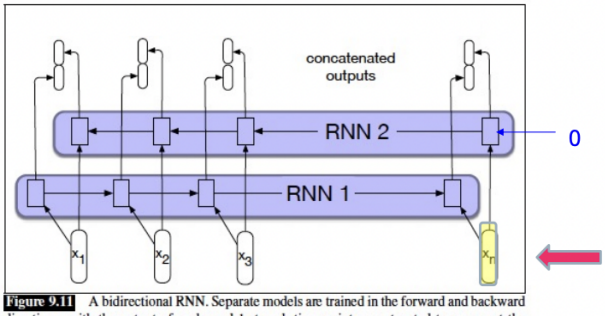
e. 

Figure 9.11 A bidirectional RNN. Separate models are trained in the forward and backward directions, with the output of each model at each time point concatenated to represent the bidirectional state at that time point.

f. These passes happen simultaneously, but let's consider them one at a time.

g. First let's consider the backwards pass through the input sequence.

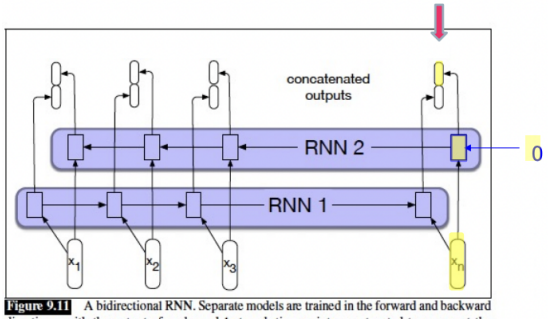
h. 

Figure 9.11 A bidirectional RNN. Separate models are trained in the forward and backward directions, with the output of each model at each time point concatenated to represent the bidirectional state at that time point.

i. Calculate the activation vector for the last backward unit. (activation is passed along and stored)

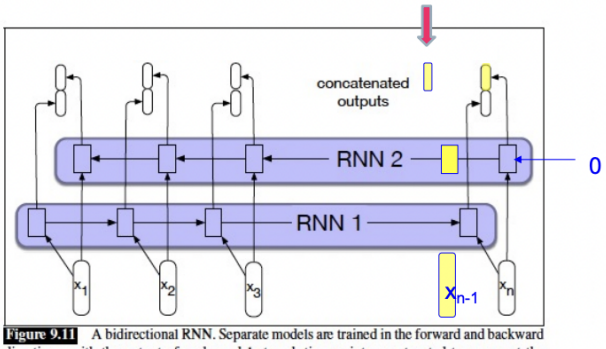


Figure 9.11 A bidirectional RNN. Separate models are trained in the forward and backward directions, with the output of each model at each time point concatenated to represent the bidirectional state at that time point.

j.

k. Continue through the sequence backwards

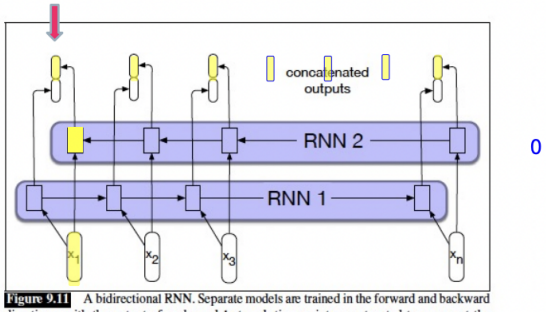


Figure 9.11 A bidirectional RNN. Separate models are trained in the forward and backward directions, with the output of each model at each time point concatenated to represent the bidirectional state at that time point.

l.

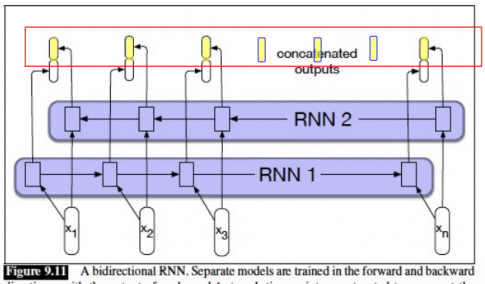


Figure 9.11 A bidirectional RNN. Separate models are trained in the forward and backward directions, with the output of each model at each time point concatenated to represent the bidirectional state at that time point.

m.

n. The backward pass activations form one part of the full activation vector.

o. Activation comes out (only half of it)

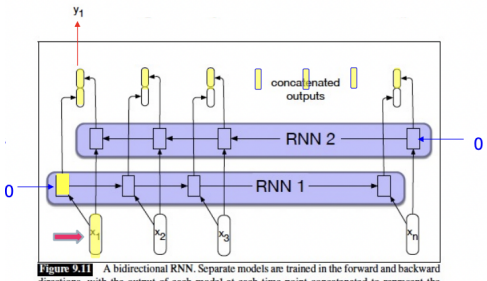


Figure 9.11 A bidirectional RNN. Separate models are trained in the forward and backward directions, with the output of each model at each time point concatenated to represent the bidirectional state at that time point.

p.

q. At the same time, the forward pass has calculated its activations.

r. Somewhere in the middle, aggregation occurs

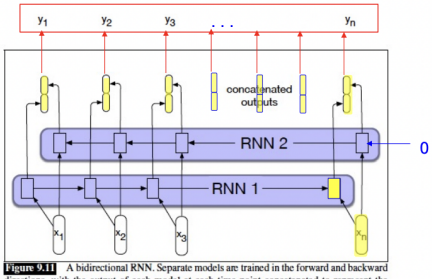
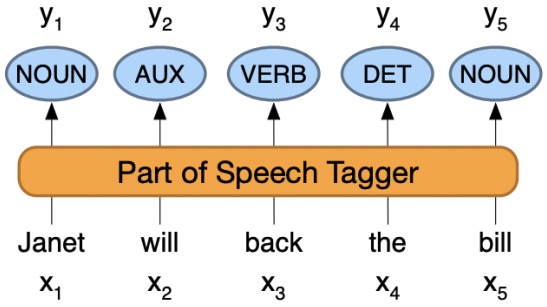


Figure 9.11 A bidirectional RNN. Separate models are trained in the forward and backward directions, with the output of each model at each time point concatenated to represent the bidirectional state at that time point.

s.

t. The concatenated activations are passed to the next layer

3. Applications of (B)RNNs: Part of Speech Tagging
    a. Map from sequence x1,…,xn of words to y1,…,yn of POS tags
    b. Sequence to another sequence (one-to-one)



c.

    d. Closed class vs. Open Class words
    e. Closed class words
        i. Relatively fixed membership, slow change over time
        ii. Usually function words: short, frequent words with grammatical function
            1. determiners: a, an, the
            2. pronouns: she, he, I
            3. prepositions: on, under, over, near, by, …
    f. Open class words
        i. Change is more rapid, especially in age of social media
        ii. Usually content words: Nouns, Verbs, Adjectives, Adverbs, Interjections, misc.
            1. Nouns: iPad, bro, meme, nothing-burger, tweet, ...
            2. Verbs: tweet, twerk, email, ...
            3. Interjections: oh, ouch, uh-huh, meh, whut, …
        iii. Abbreviations: URL, IP, WTAF, FFS, ETTD, ...
        iv. Change in class: make nouns into verbs:
            1. Add -ing to a noun: Are you any good at keyboarding?
            2. Or skip the -ing: Do you even language, bro?

g. Standard tags are from the Universal Tag Set

i.

	Tag	Description	Example
Open Class	**ADJ**	Adjective: noun modifiers describing properties	*red, young, awesome*
	ADV	Adverb: verb modifiers of time, place, manner	*very, slowly, home, yesterday*
	NOUN	words for persons, places, things, etc.	*algorithm, cat, mango, beauty*
	VERB	words for actions and processes	*draw, provide, go*
	PROPN	Proper noun: name of a person, organization, place, etc..	*Regina, IBM, Colorado*
	INTJ	Interjection: exclamation, greeting, yes/no response, etc.	*oh, um, yes, hello*
Closed Class Words	**ADP**	Adposition (Preposition/Postposition): marks a noun's spacial, temporal, or other relation	*in, on, by under*
	AUX	Auxiliary: helping verb marking tense, aspect, mood, etc.,	*can, may, should, are*
	CCONJ	Coordinating Conjunction: joins two phrases/clauses	*and, or, but*
	DET	Determiner: marks noun phrase properties	*a, an, the, this*
	NUM	Numeral	*one, two, first, second*
	PART	Particle: a preposition-like form used together with a verb	*up, down, on, off, in, out, at, by*
	PRON	Pronoun: a shorthand for referring to an entity or event	*she, who, I, others*
	SCONJ	Subordinating Conjunction: joins a main clause with a subordinate clause such as a sentential complement	*that, which*
Other	**PUNCT**	Punctuation	; , ()
	SYM	Symbols like $ or emoji	$, %
	X	Other	asdf, qwfg

h. Part of Speech (POS) Tagging is an important step in many parts of NLP.

i. POS Tagging "involves identifying and labeling each word in a sentence with its corresponding part of speech (such as nouns, verbs, adjectives, etc.), based on both its definition and its context."

j. Applications
   i. Grammar checking
   ii. Speech recognition
   iii. Search engines
   iv. Machine translation
   v. Named Entity Recognition
   vi. Text-to-speech and speech-to-text
   vii. Linguistic research and education

k. In all of these, POS Tagging can assist in resolving ambiguities in word meaning and use.
   i. Example:
      1. Give me the book. (noun)
      2. Book that flight. (verb)
      3. Lead the way. (verb) → Lead here is used as a verb
      4. Lead is heavy. (noun) → Lead here is used as a noun

l. How difficult is POS tagging in English?
   i. Roughly 15% of word types are ambiguous
      1. Hence 85% of word types are unambiguous
      2. Janet is always PROPN, hesitantly is always ADV
   ii. But those 15% tend to be very common.

iii. So ~60% of word tokens are ambiguous
iv. E.g., back
    1. earnings growth took a back/ADJ
    2. seat a small building in the back/NOUN
    3. a majority of senators back/VERB
    4. the bill enable the country to buy back/PART
    5. debt I was twenty-one back/ADV then

m. Sources of information for POS tagging
    i. Janet will back the bill
    AUX/NOUN/VERB? NOUN/VERB?
    ii. Prior probabilities of word/tag
        1. "will" is usually an AUX
    iii. Identity of neighboring words
        1. "the" means the next word is probably not a verb
        2. It can be noun or adjective → go from both directions to clarify
    iv. Morphology and wordshape:
        1. Prefixes unable: un- → ADJ
        2. Suffixes importantly: -ly → ADJ
        3. Capitalization Janet: CAP → PROPN
n. Standard algorithms for POS tagging
    i. Supervised Machine Learning Algorithms:
        1. Hidden Markov Models
        2. Conditional Random Fields (CRF)/ Maximum Entropy Markov Models (MEMM)
        3. Neural sequence models (RNNs or Transformers)
        4. Large Language Models (like BERT), finetuned
    ii. All required a hand-labeled training set, all about equal performance (97% on English)
    iii. All make use of information sources we discussed
        1. Via human created features: HMMs and CRFs
        2. Via representation learning: Neural LMs
o. For each word in the sequence, the estimated tag is compared with the actual tag label, and the log loss is added across the whole sequence; to prevent longer sequences from having lower probabilities, we take the average log loss per token:
    i. Log loss: 0.01 + 0.003 + 0.023 + 0.005 + 0.04 = 0.081 / 5 = 0.0162

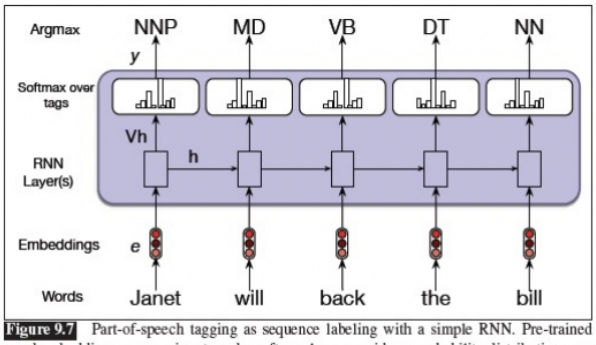
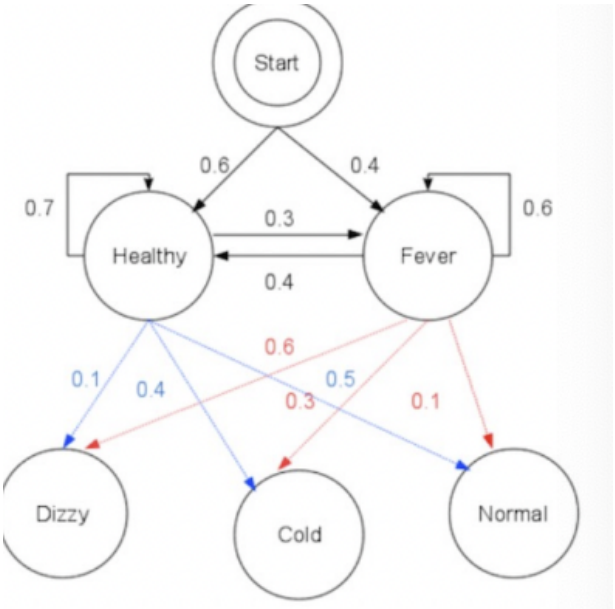


Figure 9.7 Part-of-speech tagging as sequence labeling with a simple RNN. Pre-trained word embeddings serve as inputs and a softmax layer provides a probability distribution over the part-of-speech tags as output at each time step.

ii.

iii. Sequence to Sequence and we combine the loss

4. Part of Speech Tagging with Hidden Markov Models
    a. Hidden Markov Models are Markov Chains with observations and emission probabilities. The states are considered to be unobservable or "hidden."



b.

i. Healthy and Fever are hidden

$$Q = \{1\,(Healthy),\ 2\,(Fever)\}$$

$$\pi = [0.6,\ 0.4]$$

$$O = \{1\,(Dizzy),\ 2\,(Cold),\ 3\,(Normal)\}$$

$$A = \begin{array}{|c|c|c|} \hline & 1 & 2 \\ \hline 1 & 0.7 & 0.3 \\ \hline 2 & 0.6 & 0.4 \\ \hline \end{array}$$

$$B = \begin{array}{|c|c|c|c|} \hline & 1 & 2 & 3 \\ \hline 1 & 0.1 & 0.4 & 0.5 \\ \hline 2 & 0.6 & 0.3 & 0.1 \\ \hline \end{array}$$

c.

d. You can learn the probabilities (machine learning problem)

e. There are three main problems that are studied with HMMs:

i. Evaluation problem. Given the HMM M=(A, B, p,O,B) and the observation sequence o1 o2 ... oK , calculate the probability that model M has generated the sequence.

ii. Decoding problem. Given the HMM M=(A, B, p,O,B) and the observation sequence o1 o2 ... oK , calculate the most likely sequence of hidden states s1, s2, ... sK , that produced this observation sequence.
iii. Learning problem. Given some training observation sequences o1 o2 ... oK and general structure of HMM (numbers of hidden and visible states), determine HMM parameters M=(A, B, p,O,B) that best fit the training data (alternately, determine some subset of the parameters, the others being given).

f. The decoding problem is used to do POS/NER tagging:

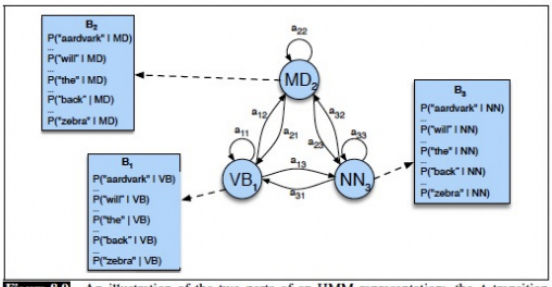


Figure 8.9 An illustration of the two parts of an HMM representation: the *A* transition probabilities used to compute the prior probability, and the *B* observation likelihoods that are associated with each state, one likelihood for each possible observation word.

g. From a training corpus of annotated sentences, determine all the parameters of an HMM model M:
i. Q = Parts of speech
ii. pi = Probabilities that each POS starts a sentence;
iii. A = Observed probabilities of bigrams P(pi | pi-1) for parts of speech pi and pi-1.
iv. O = Vocabulary
v. B = Observed probability of each word being a given POS in corpus.

h. The Viterbi or Forward algorithm, based on a 2D trellis of possible hidden states at each point in the sequence of observations, determines the most likely path, using dynamic programming.

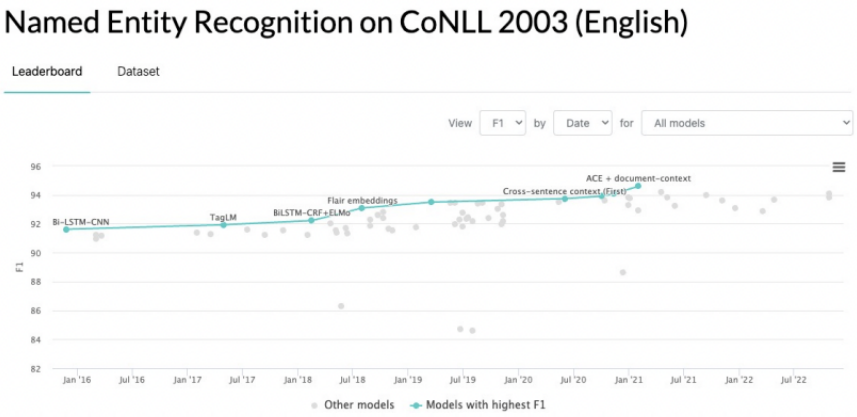
i. POS tagging performance in English
i. What is SOTA for accuracy of current approaches?
1. About 97%
a. Hasn't changed in the last 10+ years
b. HMMs, CRFs, BERT perform similarly .
c. Human accuracy about the same
ii. But baseline is 92%! So best methods only give 5% boost!
1. Baseline is performance of stupidest possible method
a. "Most frequent class baseline" is an important baseline for many tasks
i. Tag every word with its most frequent tag
ii. (and tag unknown words as nouns)

2. Partly easy because
    a. Many words are unambiguous

5. Application of (B)RNNs: Named Entity Recognition (similar but is phrases)
    a. A Named entity, in its core usage, means anything that can be referred to with a proper name. Most common 4 tags: (Capitalized)
    b. PER (Person): “Marie Curie”
    c. LOC (Location): “New York City”
    d. ORG (Organization): “Stanford University”
    e. GPE (Geo-Political Entity): "Boulder, Colorado"
    f. Often multi-word phrases
    g. But the term is also extended to things that aren't entities:
        i. dates, times, prices
    h. The task of named entity recognition (NER): (distinguishing phrases as entity)
        i. Find spans of text that constitute proper names
        ii. Tag the type of the entity.
    i. Why NER?
        i. Sentiment analysis: consumer’s sentiment toward a particular company or person.
        ii. Question Answering: answer questions about an entity.
        iii. Information Extraction: Extracting facts about entities from text.
    j. Why is NER Hard?
        i. Segmentation
            1. In POS tagging, no segmentation problem since each word gets one tag.
            2. In NER we have to find and segment the entities!
        ii. Type ambiguity
    k. How can we turn this structured problem into a sequence problem like POS tagging, with one label per word?
        i. Using BIO Tagging.
        ii. [PER Jane Villanueva] of [ORG United] , a unit of [ORG United Airlines Holding] , said the fare applies to the [LOC Chicago] route.
            1. B: token that begins a span
            2. I: tokens inside a span
            3. O: tokens outside of any span
        iii.

Words	BIO Label
Jane	B-PER
Villanueva	I-PER
of	O
United	B-ORG
Airlines	I-ORG
Holding	I-ORG
discussed	O
the	O
Chicago	B-LOC
route	O
.	O

6. SOTA for NER Tagging



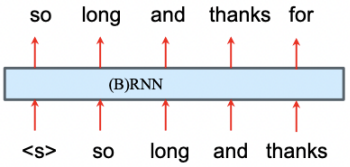
a.

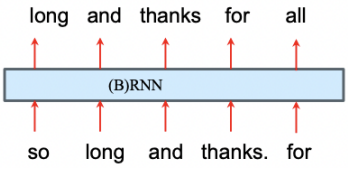
7. Applications of RNNs: Generative Language Models
   a. Recall: A Language Model assigns a probability to each sequence of words. To teach an RNN a language model, we can train it on subsequences of sentences
   b. Sentence: <s> so long and thanks for all the fish ! </s>
   c. Break into equal-length subsequences (here 5, but typically longer)

<s> so long and thanks
so long and thanks for
long and thanks for all
and thanks for all the
thanks for all the fish
for all the fish !
all the fish ! </s>

d.

e. Then we train the network on inputs and outputs





f.

   g. Training based on the sequence to sequence but use it as a vector to generate sentences (length is not the point → </s>)
   h. Becomes a vector to sequence model
   i. Generates and feeds to the next level
   j. Can give different sentence symbol → genre, etc., questions, solutions