Recurrent Neural Networks

THE UNIVERSITY OF UTAH

Overview

1. Modeling sequences
2. Recurrent neural networks: An abstraction
3. Usage patterns for RNNs
4. BiDirectional RNNs
5. A concrete example: The Elman RNN
6. The vanishing gradient problem
7. Long short-term memory units

Overview

1. *Modeling sequences*
2. Recurrent neural networks: An abstraction
3. Usage patterns for RNNs
4. BiDirectional RNNs
5. A concrete example: The Elman RNN
6. The vanishing gradient problem
7. Long short-term memory units

Sequences abound in NLP

S a l t   L a k e   C i t y

Words are sequences of characters

Sequences abound in NLP

S a l t   L a k e   C i t y

John lives in Salt Lake City

Sentences are sequences of words

Sequences abound in NLP

S a l t   L a k e   C i t y

John   lives   in   Salt   Lake   City

John lives in Salt Lake City.   He enjoys hiking with his dog.   His cat hates hiking.

Paragraphs are sequences of sentences

Sequences abound in NLP

S a l t   L a k e   C i t y

John   lives   in   Salt   Lake   City

John lives in Salt Lake City.   He enjoys hiking with his dog.   His cat hates hiking.

And so on… inputs are naturally sequences at different levels

Sequences abound in NLP

S a l t   L a k e   C i t y

John lives in Salt Lake City

John lives in Salt Lake City. He enjoys hiking with his dog. His cat hates hiking.

Outputs can also be sequences

Sequences abound in NLP

S a l t   L a k e   C i t y

John lives in Salt Lake City

John lives in Salt Lake City. He enjoys hiking with his dog. His cat hates hiking.

John lives in Salt Lake City

Part-of-speech tags form a sequence

Sequences abound in NLP

S a l t L a k e C i t y

John lives in Salt Lake City

John lives in Salt Lake City. He enjoys hiking with his dog. His cat hates hiking.

John lives in Salt Lake City

Noun Verb Preposition Noun Noun Noun

Part-of-speech tags form a sequence

Sequences abound in NLP

S a l t   L a k e   C i t y

John lives in Salt Lake City

John lives in Salt Lake City. He enjoys hiking with his dog. His cat hates hiking.

Noun Verb Preposition Noun Noun Noun

John lives in Salt Lake City

Person Location

Even things that don't look like a sequence can be made to look like one

Example: Named entity tags

Sequences abound in NLP

S a l t L a k e C i t y

John lives in Salt Lake City

John lives in Salt Lake City. He enjoys hiking with his dog. His cat hates hiking.

Noun Verb Preposition Noun Noun Noun

John lives in Salt Lake City

B-PER O O B-LOC I-LOC I-LOC

Even things that don't look like a sequence can be made to look like one

Example: Named entity tags

Sequences abound in NLP

S a l t L a k e C i t y

John lives in Salt Lake City

John lives in Salt Lake City. He enjoys hiking with his dog. His cat hates hiking.

Noun Verb Preposition Noun Noun Noun

B-PER O O B-LOC I-LOC I-LOC

And we can get very creative with such encodings

Example: We can encode parse trees as a sequence of decisions needed to construct the tree

Sequences abound in NLP

S a l t   L a k e   C i t y

John lives in Salt Lake City

Natural question: How do we model sequential inputs and outputs?

John lives in Salt Lake City. He enjoys hiking with his dog. His cat hates hiking.

Noun Verb Preposition Noun Noun Noun

B-PER O O B-LOC I-LOC I-LOC

And we can get very creative with such encodings

Example: We can encode parse trees as a sequence of decisions needed to construct the tree

Sequences abound in NLP

S a l t L a k e C i t y

John lives in Salt Lake City

Natural question: How do we model sequential inputs and outputs?

More concretely, we need a mechanism that allows us to

1. Capture sequential dependencies between inputs

2. Model uncertainty over sequential outputs

And we can get very creative with such encodings

Example: We can encode parse trees as a sequence of decisions needed to construct the tree

Modeling sequences: The problem

Suppose we want to build a language model that computes the probability of sentences

We can write the probability as

$$P(x_1, x_2, x_3, \cdots, x_n) = \prod_i P(x_i \mid x_1, x_2 \cdots, x_{i-1})$$

Example: A Language model

It was a bright cold day in April.

$$P(\text{It was a bright cold day in April}) =$$

Example: A Language model

It was a bright cold day in April.

$$P(\text{It was a bright cold day in April}) =$$

$$P(\text{It})\times$$ ← Probability of a word starting a sentence

Example: A Language model

It was a bright cold day in April.

$$P(\text{It was a bright cold day in April}) =$$

$$P(\text{It}) \times$$ ← Probability of a word starting a sentence

$$P(\text{was}|\text{It}) \times$$ ← Probability of a word following “It”

Example: A Language model

It was a bright cold day in April.

$$P(\text{It was a bright cold day in April}) =$$

$$P(\text{It})\times$$ ← Probability of a word starting a sentence

$$P(\text{was}|\text{It})\times$$ ← Probability of a word following "It"

$$P(\text{a}|\text{It was})\times$$ ← Probability of a word following "It was"

Example: A Language model

It was a bright cold day in April.

$$P(\text{It was a bright cold day in April}) =$$

$P(\text{It}) \times$ ← Probability of a word starting a sentence

$P(\text{was}|\text{It}) \times$ ← Probability of a word following "It"

$P(\text{a}|\text{It was}) \times$ ← Probability of a word following "It was"

$P(\text{bright}|\text{It was a}) \times$ ← Probability of a word following "It was a"

Example: A Language model

It was a bright cold day in April.

$$
\begin{aligned}
&P(\text{It was a bright cold day in April}) = \\
&\quad P(\text{It}) \times && \leftarrow \text{Probability of a word starting a sentence} \\
&\quad P(\text{was}|\text{It}) \times && \leftarrow \text{Probability of a word following ``It''} \\
&\quad P(\text{a}|\text{It was}) \times && \leftarrow \text{Probability of a word following ``It was''} \\
&\quad P(\text{bright}|\text{It was a}) \times && \leftarrow \text{Probability of a word following ``It was a''} \\
&\quad P(\text{cold}|\text{It was a bright}) \times \\
&\quad P(\text{day}|\text{It was a bright cold}) \times \cdots
\end{aligned}
$$

A history-based model

$$P(x_1, x_2, \cdots, x_n) = \prod_{i=1}^{n} P(x_i | x_1, x_2, \cdots, x_{i-1})$$

Each token is dependent on all the tokens that came before it

- Simple conditioning
- Each $P(x_i \mid \ldots)$ is a multinomial probability distribution over the tokens

A history-based model

$$P(x_1, x_2, \cdots, x_n) = \prod_{i=1}^{n} P(x_i | x_1, x_2, \cdots, x_{i-1})$$

Each token is dependent on all the tokens that came before it

- Simple conditioning
- Each $P(x_i \mid \ldots)$ is a multinomial probability distribution over the tokens

What is the problem here?

- How many parameters do we have?
  - Grows with the size of the sequence!

The traditional solution: Lose the history

Make a modeling assumption

Example: The first order Markov model assumes that

$$P\left(x_i \mid x_1, x_2, \cdots, x_{i-1}\right) = P(x_i \mid x_{i-1})$$

The traditional solution: Lose the history

Make a modeling assumption

Example: The first order Markov model assumes that

$$P\,(\,x_i \mid x_1, x_2, \cdots, x_{i-1}\,) = P(x_i \mid x_{i-1})$$

This allows us to simplify

$$P(x_1, x_2, x_3, \cdots, x_n) = \prod_i P(x_i \mid x_1, x_2 \cdots, x_{i-1})$$

These dependencies are ignored

The traditional solution: Lose the history

Make a modeling assumption

Example: The first order Markov model assumes that

$$P\left(x_i \mid x_1, x_2, \cdots, x_{i-1}\right) = P(x_i \mid x_{i-1})$$

This allows us to simplify

$$P(x_1, x_2, x_3, \cdots, x_n) = \prod_i P(x_i \mid x_{i-1})$$

Example: Another language model

It was a bright cold day in April

$$
\begin{aligned}
&P(\text{It was a bright cold day in April}) = \\
&\quad P(\text{It}) \times \\
&\quad P(\text{was}|\text{It}) \times \\
&\quad P(\text{a}|\text{was}) \times \\
&\quad P(\text{bright}|\text{a}) \times \\
&\quad P(\text{cold}|\text{bright}) \times \\
&\quad P(\text{day}|\text{cold}) \times \cdots
\end{aligned}
$$

Probability of a word starting a sentence

Probability of a word following "It"

Probability of a word following "was"

Probability of a word following "a"

Example: Another language model

It was a bright cold day in April

$$P(\text{It was a bright cold day in April}) =$$
$$P(\text{It})\times$$ ← Probability of a word starting a sentence
$$P(\text{was}|\text{It})\times$$ ← Probability of a word following "It"
$$P(\text{a}|\text{was})\times$$ ← Probability of a word following "was"
$$P(\text{bright}|\text{a})\times$$ ← Probability of a word following "a"
$$P(\text{cold}|\text{bright})\times$$
$$P(\text{day}|\text{cold})\times\cdots$$

If there are K tokens/states, how many parameters do we need?

Example: Another language model

It was a bright cold day in April

$$
\begin{aligned}
&P(\text{It was a bright cold day in April}) = \\
&\quad P(\text{It}) \times \\
&\quad P(\text{was}|\text{It}) \times \\
&\quad P(\text{a}|\text{was}) \times \\
&\quad P(\text{bright}|\text{a}) \times \\
&\quad P(\text{cold}|\text{bright}) \times \\
&\quad P(\text{day}|\text{cold}) \times \cdots
\end{aligned}
$$

- $P(\text{It})$ ← Probability of a word starting a sentence
- $P(\text{was}|\text{It})$ ← Probability of a word following "It"
- $P(\text{a}|\text{was})$ ← Probability of a word following "was"
- $P(\text{bright}|\text{a})$ ← Probability of a word following "a"

If there are K tokens/states, how many parameters do we need? $O(K^2)$

Can we do better?

- Can we capture the meaning of the entire history without arbitrarily growing the number of parameters?

- Or equivalently, can we discard the Markov assumption?

Can we do better?

- Can we capture the meaning of the entire history without arbitrarily growing the number of parameters?
- Or equivalently, can we discard the Markov assumption?
- Can we represent arbitrarily long sequences as fixed sized vectors?
  - Perhaps to provide features for subsequent classification

Can we do better?

- Can we capture the meaning of the entire history without arbitrarily growing the number of parameters?

- Or equivalently, can we discard the Markov assumption?

- Can we represent arbitrarily long sequences as fixed sized vectors?
  - Perhaps to provide features for subsequent classification

- Answer: Recurrent neural networks (RNNs)