Week 02: Word Embeddings

---

**Logistics**

- HW1 is due on Thursday
- Submit on gradescope
  - If you worked in a group, create a group and then submit
- Clarifications are on Ed
- Come to office hours if you have questions

---

**MLPs:**
- Fully connected layers
- Require more parameters and computational resources
- Flexible and can handle various input types

**CNNs:**
- Convolutional layers with filters
- Designed specifically for structured input like images
- Inherently translation invariant due to shared weights
- Requires fewer parameters

---

How to handle text data?

Language Modeling: predict the next word

Assign probabilities to text.

Given a sequence \( \{x_1, x_2, \ldots, x_T\} \), we want to maximize \( P(x_1, x_2, \ldots, x_T) \).

\[
P(x_1, x_2, \ldots, x_T) = P(x_1)P(x_2|x_1)P(x_3|x_1, x_2)P(x_4|x_1, x_2, x_3) \ldots P(x_T|x_1, x_2, \ldots x_{T-1})
\]

\( P(I \text{ like cats because they look cute}) = P(I)P(like | I)P(cats | I \text{ like}) P(as | I \text{ like cats}) P(they | I \text{ like cats because}) P(look | they) P(cute | look) \)

Predict the next word given current text!

\( n \)-Gram Language Model

\( n \)-Gram: chunk of \( n \) consecutive words

Uni-gram: "I" "like" "cats" "as" "they" "look" "cute"

Bi-gram: "I like" "like cats" "cats as" "as they" ...

Tri-gram: "I like cats" "like cats as" "cats as they" ...

Count the frequency of each \( n \)-grams and predict next word!

Assume each word only depends on previous \( n - 1 \) words.

\[
P(x_T|x_1, \ldots, x_{T-1}) = \frac{P(x_T|x_{T-n+1}, \ldots, x_{T-1})}{\text{count}(x_{T-n+1}, \ldots, x_{T-1})}
\]

In bi-gram LM

\( P(I \text{ like cats as they look cute}) = P(I)P(like | I)P(cats | I \text{ like}) P(as | cats) P(they | because) P(look | they) P(cute | look) \)
Discuss: Do you want to have a large $n$ or a small $n$ in a $n$-gram model?

$n$-Gram Language Model: issue

- **Uni-gram**: “I” “like” “cats” “as” “they” “look” “cute”
- **Bi-gram**: “I like” “like cats” “cats as” “as they” …
- **Tri-gram**: “I like cats” “like cats as” “cats as they” …

$n$-Gram: chunk of $n$ consecutive words

Count the frequency of each $n$-grams and predict next word!

Assume each word only depends on previous $n - 1$ words.

$$P(x_t|x_{t-n+1}, \ldots, x_{t-1}) = \frac{\text{count}(x_t, x_{t-n+1}, \ldots, x_{t-1})}{\text{count}(x_{t-n+1}, \ldots, x_{t-1})}$$

Increase $n$ provides contextual information, but exponentially increase the size of the counting table!

Bag of Words

What are word embeddings

- **What are Word Embeddings?**
  - vector representations of words that capture semantic relationships

https://koushik1102.medium.com/nlp-bag-of-words-and-tf-idf-explained-64f14d0a4f55
Semantic similarity

- Motivation
  - Put words into vectors so we can measure the similarity between words
  - Use cosine similarity

Why Do We Need Word Embeddings?

- Numerical Input
- Shows Similarity and Distance

What does ong choi mean?

Suppose you see these sentences:
- Ong choi is delicious sautéed with garlic.
- Ong choi is superb over rice
- Ong choi leaves with salty sauces

And you've also seen these:
- ...spinach sautéed with garlic over rice
- Chard stems and leaves are delicious
- Collard greens and other salty leafy greens

https://web.stanford.edu/~jurafsky/slp3/
Word2Vec

- We want vectors for words so that the context of a word can suggest the vector of this word, and vice versa
- Idea: **Similar words appear in similar contexts**

A cup of **coffee** is on the table. **Coffee** helps me focus. Espresso is my favorite type of **coffee**.

---

Word2Vec - Training

**SkipGram** - Predict context from target

A cup of **coffee** is on the table.

**Continuous Bag of Words (CBOW)** - predict target from context

SkipGram - Training samples

A cup of **coffee** is on the table.

(coffee, cup)

(coffee, of)

(coffee, is)

(coffee, on)
Discuss: Word2Vec Architecture - CBOW

What is the output of multiplying the one-hot vector [0,1,0,0,0,0] with \( W \)?

Looking closer…

- We observe that every row of the \( W \) matrix corresponds to a target word and every column of the \( W' \) matrix corresponds to a context word.
- We compute the probability of a target-context pair as:

\[
p(w_c | w_t) = \frac{\exp(W_{ct}W_{t}^T)}{\sum_{i=1}^{|V|} \exp(W_{ci}W_{t}^T)}
\]
Word2Vec Architecture - SkipGram

Predict every target word from each context word!

cup $\rightarrow$ coffee

Word2Vec Architecture - CBOW

(cup, of, is, on) $\rightarrow$ coffee

Cross Entropy

- Cross Entropy: lower cross entropy indicates high similarity between two distributions

$$\mathcal{L}_\theta = -\sum_{i=1}^{|V|} y_i \log p(w_i | w_t) = -\log p(w_c | w_t)$$

- So the loss function is:

$$\mathcal{L}_\theta = -\log \frac{\exp(W_c\theta)}{\sum_{i=1}^{|V|} \exp(W_i\theta)}$$

Where do we get the word embeddings from in this version of Word2Vec (CBOW)?
X 2 vec

- Generate vector representations (embeddings) for various data types
- Examples:
  - Word2Vec
  - Doc2Vec
  - Node2Vec
  - Item2Vec
  - Sent2Vec

Demo

Visualize: https://projector.tensorflow.org/
Explore: http://epsilon-it.utu.fi/wv_demo/

Doc2Vec

- A vector to represent a paragraph, regardless of length
  - embeddings for paragraph and words
  - Applications: Document classification, sentiment analysis, recommendation systems, and information retrieval

In vector space...

woman
man
queen
king
Word embeddings are time-dependent (why?)

- Semantic similarity of words depends on time.

Problems with word2vec

- Words with multiple meanings only have one representation
  - eg. bank of river or bank of money
  - Need contextual information
- Limited Context
  - only trained on words within the context window

How to use word vectors with neural networks?

- Inputs and outputs don't have fixed lengths
- Features are not shared

Let's simplify!

What if we have a single word and a single output?
Recurrent neural network (RNN)

Parameterize RNN

- Too many parameters if we have a long sequence!
- Longer sequence parameters will not receive many updates

RNN w/ parameter-sharing

Simple fix: use the same parameters across different timesteps.

Discuss: RNN w/ parameter-sharing

Simple fix: use the same parameters across different timesteps.

A non-linearity is applied to the output of the recurrent unit before it is passed to the next time step or to the output layer of the network.

Write a closed-form solution for $h_t$ and $\hat{y}_t$.
Types of RNNs

- **one to one**
- **one to many**
- **many to one**
- **many to many**


Stacking RNN Layers

- Unfold a recurrent neural network in time
- Gradients are accumulated across all time steps by applying the chain rule
- Propagate gradients backwards through time steps

Sequence Timesteps

Backpropagation through the Time (BPTT)
Backpropagation through the Time (BPTT)

\[
\frac{\partial C}{\partial w_{th}} = \frac{\partial C}{\partial y_T} \cdot \frac{\partial y_T}{\partial h_T} \cdot \frac{\partial h_T}{\partial h_{T-1}} \cdot \frac{\partial h_{T-1}}{\partial w_{th}} = \sum_{k=1}^{T} \frac{\partial C}{\partial y_T} \cdot \frac{\partial y_T}{\partial h_T} \cdot \frac{\partial h_T}{\partial h_k} \cdot \frac{\partial h_k}{\partial w_{th}} \cdot (\prod_{j=k}^{T-1} \frac{\partial h_j}{\partial h_{T-1}})
\]

Recap

- N-gram models
- Bag-of-words representations
- Word2Vec
  - CBOW: use context to predict target word
  - SkipGram: use target word to predict context
- RNN
  - Has an internal state (memory)
  - Can handle arbitrary sequences of inputs
  - Trained with back propagation through time

Image credits:
https://lilianweng.github.io/posts/2017-10-15-word-embedding/