Topics in AI (CPSC 532S): Multimodal Learning with Vision, Language and Sound

Lecture 8: Word2Vec, Language Models and RNNs
Course Logistics

- Assignment 3
- Final project group Goolge form will be out tomorrow
Representing a **Word**: One Hot Encoding

<table>
<thead>
<tr>
<th>Vocabulary</th>
<th>One-hot encodings</th>
</tr>
</thead>
<tbody>
<tr>
<td>dog</td>
<td>[1, 0, 0, 0, 0, 0, 0, 0, 0, 0]</td>
</tr>
<tr>
<td>cat</td>
<td>[0, 1, 0, 0, 0, 0, 0, 0, 0, 0]</td>
</tr>
<tr>
<td>person</td>
<td>[0, 0, 1, 0, 0, 0, 0, 0, 0, 0]</td>
</tr>
<tr>
<td>holding</td>
<td>[0, 0, 0, 1, 0, 0, 0, 0, 0, 0]</td>
</tr>
<tr>
<td>tree</td>
<td>[0, 0, 0, 0, 1, 0, 0, 0, 0, 0]</td>
</tr>
<tr>
<td>computer</td>
<td>[0, 0, 0, 0, 0, 1, 0, 0, 0, 0]</td>
</tr>
<tr>
<td>using</td>
<td>[0, 0, 0, 0, 0, 0, 1, 0, 0, 0]</td>
</tr>
</tbody>
</table>

*slide from V. Ordonex*
Representing **Phrases**: Bag-of-Words

<table>
<thead>
<tr>
<th>Phrase</th>
<th>Bag-of-Words Representation</th>
</tr>
</thead>
<tbody>
<tr>
<td>person holding dog</td>
<td>{3, 4, 1}</td>
</tr>
<tr>
<td>person holding cat</td>
<td>{3, 4, 2}</td>
</tr>
<tr>
<td>person using computer</td>
<td>{3, 7, 6}</td>
</tr>
<tr>
<td>person using computer person holding cat</td>
<td>{3, 3, 7, 6, 2}</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Vocabulary</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>dog</td>
<td>1</td>
</tr>
<tr>
<td>cat</td>
<td>2</td>
</tr>
<tr>
<td>person</td>
<td>3</td>
</tr>
<tr>
<td>holding</td>
<td>4</td>
</tr>
<tr>
<td>tree</td>
<td>5</td>
</tr>
<tr>
<td>computer</td>
<td>6</td>
</tr>
<tr>
<td>using</td>
<td>7</td>
</tr>
</tbody>
</table>

*slide from V. Ordonex*
— At least certain aspects of the meaning of lexical expressions depend on their distributional properties in the linguistic contexts

— The degree of semantic similarity between two linguistic expressions is a function of the similarity of the two linguistic contexts in which they can appear
What is the meaning of “bardiwac”? 

— He handed her glass of bardiwac.
— Beef dishes are made to complement the bardiwacs.
— Nigel staggered to his feet, face flushed from too much bardiwac.
— Malbec, one of the lesser-known bardiwac grapes, responds well to Australia’s sunshine.
— I dined off bread and cheese and this excellent bardiwac.
— The drinks were delicious: blood-red bardiwac as well as light, sweet Rhenish.

**bardic** is an alcoholic beverage made from grapes

* Adopted from slides by Louis-Philippe Morency*
The **Use Theory** of Meaning

“If you can understand and predict in which context a word will appear in, then you understood the meaning of the word”  [Paul Horwich]
Geometric Interpretation: Co-occurrence as feature

— Row vector describes usage of word in a corpus of text

— Can be seen as coordinates of the point in an n-dimensional Euclidean space

Co-occurrence Matrix

* Slides from Louis-Philippe Morency
Distance and Similarity

— Illustrated in two dimensions

— Similarity = spatial proximity (Euclidian distance)

— Location depends on frequency of noun (dog is 27 times as frequent as cat)

* Slides from Louis-Philippe Morency
Angle and Similarity

- direction is more important than location
- normalize length of vectors
- or use angle as a distance measure

* Slides from Louis-Philippe Morency
Geometric Interpretation: Co-occurrence as feature

- Row vector describes usage of word in a corpus of text
- Can be seen as coordinates of the point in an n-dimensional Euclidean space

Co-occurrence Matrix

Way too high dimensional!
SVD for Dimensionality Reduction

\[
X = U \Sigma \hat{V}^T
\]

*slide from Vagelis Hristidis*
Learned Word Vector Visualization

We can also use other methods, like LLE here:


[ Roweis and Saul, 2000 ]
Issues with **SVD**

**Computational** cost for a $d \times n$ matrix is $\mathcal{O}(dn^2)$, where $d < n$

- Makes it not possible for large number of word vocabularies or documents

It is hard to incorporate out of sample *(new)* words or documents
**word2vec**: Representing the Meaning of Words

**Key idea**: Predict surrounding words of every word

**Benefits**: Faster and easier to incorporate new document, words, etc.

*slide from Vagelis Hristidis [Mikolov et al., 2013]*
**word2vec**: Representing the Meaning of Words

**Key idea**: Predict surrounding words of every word

**Benefits**: Faster and easier to incorporate new document, words, etc.

Continuous Bag of Words (**CBO**W): use context words in a window to predict middle word

**Skip-gram**: use the middle word to predict surrounding ones in a window

---

*slide from Vagelis Hristidis [Mikolov et al., 2013]*
**CBOW**: Continuous Bag of Words

**Example**: “The cat sat on floor” (window size 2)

*slide from Vagelis Hristidis [Mikolov et al., 2013]*
CBOW: Continuous Bag of Words

*slide from Vagelis Hristidis*
CBOW: Continuous Bag of Words

Input layer

\[
x \in \mathbb{R}^{|V|}
\]

Hidden layer

\[
\hat{v} \in \mathbb{R}^{N}
\]

Output layer

\[
\hat{y} \in \mathbb{R}^{|V|}
\]

\[
w_{|V| \times |N|}
\]

\[
w'_{|N| \times |V|}
\]

\[
\text{cat}
\]

\[
on
\]

\[
sat
\]

[ Mikolov et al., 2013 ]

*slide from Vagelis Hristidis
**CBOW**: Continuous Bag of Words

- **Input** layer
- **Hidden** layer
- **Output** layer

Parameters to be learned

- $x \in \mathbb{R}^{|V|}$
- $\hat{v} \in \mathbb{R}^{N}$
- $\hat{y} \in \mathbb{R}^{|V|}$

- $W_{|V| \times |N|}$
- $W'_{|N| \times |V|}$

*slide from Vagelis Hristidis

[ Mikolov et al., 2013 ]
**CBOW**: Continuous Bag of Words

- **Input layer**
- **Hidden layer**
- **Output layer**

**Parameters** to be learned

- $x \in \mathbb{R}^{|V|}$
- $v \in \mathbb{R}^N$
- $\hat{y} \in \mathbb{R}^{|V|}$
- $W_{|V| \times |N|}$
- $W'_{|N| \times |V|}$

**Size** of the word vector (e.g., 300)

*slide from Vagelis Hristidis [Mikolov et al., 2013]*
CBOW: Continuous Bag of Words

[ Mikolov et al., 2013 ]

*slide from Vagelis Hristidis
**CBOW**: Continuous Bag of Words

\[
\mathbf{W}^T_{|V| \times |N|} \times \mathbf{x}_{cat} = \mathbf{v}_{cat}
\]

*slide from Vagelis Hristidis [Mikolov et al., 2013]*
**CBOW: Continuous Bag of Words**

- **Input layer**

\[ x_{cat} \times W_{|V| \times |N|} \times x_{on} = v_{on} \]

\[
\begin{array}{cccccccc}
0.1 & 2.4 & 1.6 & 1.8 & 0.5 & 0.9 & \ldots & \ldots & 3.2 \\
0.5 & 2.6 & 1.4 & 2.9 & 1.5 & 3.6 & \ldots & \ldots & 6.1 \\
\ldots & \ldots & \ldots & \ldots & \ldots & \ldots & \ldots & \ldots & \ldots \\
\ldots & \ldots & \ldots & \ldots & \ldots & \ldots & \ldots & \ldots & \ldots \\
0.6 & 1.8 & 2.7 & 1.9 & 2.4 & 2.0 & \ldots & \ldots & 1.2 \\
\end{array}
\]

*slide from Vagelis Hristidis* [Mikolov et al., 2013]
**CBOW:** Continuous Bag of Words

![Diagram](image)

- **Input layer**
  - Input: \( x_{\text{cat}} \) and \( x_{\text{on}} \)
  - Weights: \( W \)
  - Output: \( x \in \mathbb{R}^{|V|} \)

- **Hidden layer**
  - Output: \( \hat{v} = \frac{v_{\text{cat}} + v_{\text{on}}}{2} \)
  - \( \hat{v} \in \mathbb{R}^{|N|} \)

- **Output layer**
  - Output: \( \hat{y} \in \mathbb{R}^{|V|} \)
  - \( \hat{y} \) is the predicted output for the word at position \( \hat{v} \)

*slide from Vagelis Hristidis

[ Mikolov et al., 2013 ]
**CBOW:** Continuous Bag of Words

\[
\begin{align*}
\text{Input layer} & \quad W_{|N| \times |V|} \times x_{\text{cat}} = v_{\text{cat}} \\
\text{Hidden layer} & \quad \hat{v} \in \mathbb{R}^{|N|} \\
\text{Output layer} & \quad W'_{|V| \times |N|} \times \hat{v} = z \\
& \quad \hat{y} = \text{softmax}(z)
\end{align*}
\]

*slide from Vagelis Hristidis, [Mikolov et al., 2013]*
**CBOW: Continuous Bag of Words**

**Input layer**

\[ x_{\text{cat}} \times x_{\text{cat}} = v_{\text{cat}} \]

\[ x_{\text{on}} \times x_{\text{on}} = v_{\text{on}} \]

**Hidden layer**

\[ v \in \mathbb{R}^{|N|} \]

**Output layer**

\[ \hat{y} = \text{softmax}(z) \]

\[ \hat{y}_{\text{sat}} \]

Optimize to get close to 1-hot encoding

*slide from Vagelis Hristidis

[ Mikolov et al., 2013 ]
CBOW: Continuous Bag of Words

\[
\begin{align*}
\mathbf{W}^{T}_{|V| \times |N|} & \\
0.1 & 2.4 & 1.6 & 1.8 & 0.5 & 0.9 & \ldots & \ldots & \ldots & 3.2 \\
0.5 & 2.6 & 1.4 & 2.9 & 1.5 & 3.6 & \ldots & \ldots & \ldots & 6.1 \\
\ldots & \ldots & \ldots & \ldots & \ldots & \ldots & \ldots & \ldots & \ldots & \ldots \\
\ldots & \ldots & \ldots & \ldots & \ldots & \ldots & \ldots & \ldots & \ldots & \ldots \\
0.6 & 1.8 & 2.7 & 1.9 & 2.4 & 2.0 & \ldots & \ldots & \ldots & 1.2 \\
\end{align*}
\]

\[
\hat{\mathbf{y}} = \text{softmax}(\mathbf{z})
\]

\[
\hat{\mathbf{y}}_{\text{sat}}
\]

*slide from Vagelis Hristidis [ Mikolov et al., 2013 ]
CBOW: Interesting Observation

There are two representations for same word!
**CBOw: Interesting Observation**

**Another way to look at it:** Maximize similarity between context word representation and the word representation itself

\[
p(w|c) = \frac{\exp \left[ (\sum_c Wx_c)^T (Wx_w) \right]}{\sum_{i \in V} \exp \left[ (Wx_i)^T (Wx_w) \right]} \]
**CBOW:** Interesting Observation

**Another way to look at it:** Maximize similarity between context word representation and the word representation itself

\[
J(W) = -\frac{1}{T} \sum_{t=1}^{T} \sum_{-m \leq j \leq m; j \neq 0} \log p(w_{t+j} | w_t)
\]

\[
p(w_{t+j} | w_t) = \frac{\exp(w_{t+j}^T w_t)}{\sum_{i=1}^{|V|} \exp(w_i^T w_t)}
\]

[ Mikolov et al., 2013 ]
Skip-Gram Model

[ Mikolov et al., 2013 ]
— **CBOw** is not great for rare words and typically needs less data to train

— **Skip-gram** better for rate words and needs more data to train the model

<table>
<thead>
<tr>
<th>Model</th>
<th>Vector Dimensionality</th>
<th>Training words</th>
<th>Accuracy [%]</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td>Semantic</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Collobert-Weston NNLM</td>
<td>50</td>
<td>660M</td>
<td>9.3</td>
</tr>
<tr>
<td>Turian NNLM</td>
<td>50</td>
<td>37M</td>
<td>1.4</td>
</tr>
<tr>
<td>Turian NNLM</td>
<td>200</td>
<td>37M</td>
<td>1.4</td>
</tr>
<tr>
<td>Mnih NNLM</td>
<td>50</td>
<td>37M</td>
<td>1.8</td>
</tr>
<tr>
<td>Mnih NNLM</td>
<td>100</td>
<td>37M</td>
<td>3.3</td>
</tr>
<tr>
<td>Mikolov RNML</td>
<td>80</td>
<td>320M</td>
<td>4.9</td>
</tr>
<tr>
<td>Mikolov RNML</td>
<td>640</td>
<td>320M</td>
<td>8.6</td>
</tr>
<tr>
<td>Huang NNLM</td>
<td>50</td>
<td>990M</td>
<td>13.3</td>
</tr>
<tr>
<td>Our NNLM</td>
<td>20</td>
<td>6B</td>
<td>12.9</td>
</tr>
<tr>
<td>Our NNLM</td>
<td>50</td>
<td>6B</td>
<td>27.9</td>
</tr>
<tr>
<td>Our NNLM</td>
<td>100</td>
<td>6B</td>
<td>34.2</td>
</tr>
<tr>
<td>CBOw</td>
<td>300</td>
<td>783M</td>
<td>15.5</td>
</tr>
<tr>
<td>Skip-gram</td>
<td>300</td>
<td>783M</td>
<td>50.0</td>
</tr>
</tbody>
</table>

[ Mikolov et al., 2013 ]
Interesting Results: **Word Analogies**

Test for linear relationships, examined by Mikolov et al. (2014)

\[ d = \arg \max_x \frac{(w_b - w_a + w_c)^T w_x}{\|w_b - w_a + w_c\|} \]

\[
\begin{align*}
\text{man:woman} & \quad \text{king:}% \\
+ \text{king} & \quad [0.30 \ 0.70] \\
- \text{man} & \quad [0.20 \ 0.20] \\
+ \text{woman} & \quad [0.60 \ 0.30] \\
\hline \\
\text{queen} & \quad [0.70 \ 0.80]
\end{align*}
\]
Interesting Results: **Word Analogies**

[ Mikolov et al., 2013 ]
Language Models

Model the **probability of a sentence**; ideally be able to sample plausible sentences.

Why is this useful?

\[
\begin{align*}
\text{arg max } P(\text{wordsequence} | \text{acoustics}) &= \frac{P(\text{acoustics} | \text{wordsequence}) \times P(\text{wordsequence})}{P(\text{acoustics})} \\
\text{arg max } P(\text{acoustics} | \text{wordsequence}) \times P(\text{wordsequence})
\end{align*}
\]

* Slides from Louis-Philippe Morency
Simple **Language Models**: N-Grams

Given a word sequence: \( w_{1:n} = [w_1, w_2, \ldots, w_n] \)

We want to estimate \( p(w_{1:n}) \)

Using **Chain Rule** of probabilities:

\[
p(w_{1:n}) = p(w_1)p(w_2|w_1)p(w_3|w_1, w_2) \cdots p(w_n|w_{1:n-1})
\]

**Bi-gram** Approximation:

\[
p(w_{1:n}) = \prod_{k=1}^{n} p(w_k|w_{k-1})
\]

**N-gram** Approximation:

\[
p(w_{1:n}) = \prod_{k=1}^{n} p(w_k|w_{k-N+1:k-1})
\]

* Slides from Louis-Philippe Morency*
Estimating Probabilities

N-gram conditional probabilities can be estimated based on raw concurrence counts in the observed sequences

**Bi-gram:**

\[ p(w_n|w_{n-1}) = \frac{C(w_{n-1}w_n)}{C(w_{n-1})} \]

**N-gram:**

\[ p(w_n|w_{n-N-1:n-1}) = \frac{C(w_{n-N-1:n-1}w_n)}{C(w_{n-N-1:n-1})} \]

* Slides from Louis-Philippe Morency*
Neural-based Unigram Language Mode

Problem: Does not model sequential information (too local)

We need sequence modeling!

* Slides from Louis-Philippe Morency
Sequence Modeling

Input

Image Maps

Convolutions

Subsampling

Fully Connected

Output
Why Model **Sequences**?

* slide from Dhruv Batra

Image Credit: Alex Graves and Kevin Gimpel
Multi-modal tasks

[ Vinyals et al., 2015 ]

A group of people shopping at an outdoor market.
There are many vegetables at the fruit stand.
Sequences where you don’t expect them …

Classify images by taking a series of “glimpses”

[ Gregor et al., ICML 2015 ]
[ Mnih et al., ICLR 2015 ]
Sequences in Inputs or Outputs?

Input: No sequence
Output: No seq.

Example:
“standard” classification / regression problems

* slide from Fei-Dei Li, Justin Johnson, Serena Yeung, cs231n Stanford
Sequences in Inputs or Outputs?

one to one

Input: No sequence
Output: No seq.
Example: “standard” classification / regression problems

one to many

Input: No sequence
Output: Sequence
Example: Im2Caption

* slide from Fei-Dei Li, Justin Johnson, Serena Yeung, cs231n Stanford
Sequences in Inputs or Outputs?

One to one

Input: No sequence
Output: No seq.
Example: “standard” classification / regression problems

One to many

Input: No sequence
Output: Sequence
Example: Im2Caption

Many to one

Input: Sequence
Output: No seq.
Example: sentence classification, multiple-choice question answering

* slide from Fei-Dei Li, Justin Johnson, Serena Yeung, cs231n Stanford
Sequences in Inputs or Outputs?

**Input:** No sequence  
**Output:** No seq.  
**Example:** “standard” classification / regression problems

**Input:** No sequence  
**Output:** Sequence  
**Example:** Im2Caption

**Input:** Sequence  
**Output:** No seq.  
**Example:** sentence classification, multiple-choice question answering

**Input:** Sequence  
**Output:** Sequence  
**Example:** machine translation, video captioning, open-ended question answering, video question answering

* slide from Fei-Dei Li, Justin Johnson, Serena Yeung, cs231n Stanford
Key Conceptual Ideas

Parameter Sharing
— in computational graphs = adding gradients

“Unrolling”
— in computational graphs with parameter sharing

Parameter Sharing + “Unrolling”
— Allows modeling arbitrary length sequences!
— Keeps number of parameters in check

* slide from Dhruv Batra
Recurrent Neural Network

* slide from Fei-Dei Li, Justin Johnson, Serena Yeung, cs231n Stanford
Recurrent Neural Network

usually want to predict a vector at some time steps

* slide from Fei-Dei Li, Justin Johnson, Serena Yeung, cs231n Stanford
Recurrent Neural Network

We can process a sequence of vectors $x$ by applying a recurrence formula at every time step:

$$h_t = f_W(h_{t-1}, x_t)$$

some function with parameters $W$

old state

new state

input vector at some time step

* slide from Fei-Dei Li, Justin Johnson, Serena Yeung, cs231n Stanford
Recurrent Neural Network

We can process a sequence of vectors $\mathbf{x}$ by applying a recurrence formula at every time step:

$$h_t = f_W(h_{t-1}, x_t)$$

Note: the same function and the same set of parameters are used at every time step.

* slide from Fei-Dei Li, Justin Johnson, Serena Yeung, cs231n Stanford
(Vanilla) **Recurrent Neural Network**

\[ y_t = W_{hy} h_t + b_y \]

\[ h_t = f_W(h_{t-1}, x_t) \]

\[ h_t = \tanh(W_{hh} h_{t-1} + W_{xh} x_t + b_h) \]

* slide from Fei-Dei Li, Justin Johnson, Serena Yeung, **cs231n Stanford**
RNN Computational Graph

\[
\begin{align*}
    h_0 & \rightarrow f_W \rightarrow h_1 \\
    X_1 & \rightarrow f_W
\end{align*}
\]

* slide from Fei-Dei Li, Justin Johnson, Serena Yeung, cs231n Stanford
RNN Computational Graph

\[ h_0 \rightarrow f_W \rightarrow h_1 \rightarrow f_W \rightarrow h_2 \]

\[ x_1 \]

\[ x_2 \]

* slide from Fei-Dei Li, Justin Johnson, Serena Yeung, cs231n Stanford
RNN Computational Graph

* slide from Fei-Dei Li, Justin Johnson, Serena Yeung, cs231n Stanford
RNN Computational Graph

Re-use the same weight matrix at every time-step

* slide from Fei-Dei Li, Justin Johnson, Serena Yeung, cs231n Stanford
RNN Computational Graph: Many to Many

* slide from Fei-Dei Li, Justin Johnson, Serena Yeung, cs231n Stanford
RNN Computational Graph: Many to Many

* slide from Fei-Dei Li, Justin Johnson, Serena Yeung, cs231n Stanford
RNN Computational Graph: Many to Many

\[ \begin{align*}
    h_0 & \rightarrow f_W & h_1 & \rightarrow f_W & h_2 & \rightarrow f_W & h_3 & \rightarrow \ldots & h_T \\
    x_1 & \rightarrow W & x_2 & \rightarrow \ldots & x_3 & \rightarrow \ldots
\end{align*} \]

* slide from Fei-Dei Li, Justin Johnson, Serena Yeung, cs231n Stanford
RNN Computational Graph: Many to One

\[ h_0 \xrightarrow{f_W} h_1 \xrightarrow{f_W} h_2 \xrightarrow{f_W} h_3 \rightarrow \ldots \rightarrow h_T \]

* slide from Fei-Dei Li, Justin Johnson, Serena Yeung, cs231n Stanford
RNN **Computational Graph**: One to Many

* slide from Fei-Dei Li, Justin Johnson, Serena Yeung, cs231n Stanford
**Sequence to Sequence**: Many to One + One to Many

**Many to one**: Encode input sequence in a single vector

**One to many**: Produce output sequence from single input vector

* slide from Fei-Dei Li, Justin Johnson, Serena Yeung, cs231n Stanford
Example: Character-level Language Model

Vocabulary:

[‘h’, ‘e’, ‘l’, ‘o’]

Example training sequence:

“hello”
Example: Character-level Language Model

Vocabulary:
['h', 'e', 'l', 'o']

Example training sequence:
“hello”

\[ h_t = \tanh(W_{hh}h_{t-1} + W_{xh}x_t + b_h) \]

* slide from Fei-Dei Li, Justin Johnson, Serena Yeung, cs231n Stanford
Example: Character-level Language Model

Vocabulary:
[‘h’, ‘e’, ‘l’, ‘o’]

Example training sequence:
“hello”
**Example:** Character-level Language Model (Sampling)

**Vocabulary:**

```
['h', 'e', 'l', 'o']
```

At test time sample one character at a time and feed back to the model
**Example:** Character-level Language Model (Sampling)

**Vocabulary:**

`['h', 'e', 'l', 'o']`

At test time sample one character at a time and feed back to the model.

* slide from Fei-Dei Li, Justin Johnson, Serena Yeung, cs231n Stanford
Example: Character-level Language Model (Sampling)

Vocabulary: ['h', 'e', 'l', 'o']

At test time sample one character at a time and feed back to the model.

* slide from Fei-Dei Li, Justin Johnson, Serena Yeung, cs231n Stanford
Example: Character-level Language Model (Sampling)

Vocabulary:
[‘h’, ‘e’, ‘l’, ‘o’]

At test time sample one character at a time and feed back to the model.
Teacher Forcing

non-teacher forcing

teacher forcing
Sampling vs. ArgMax

**Sampling**: allows to generate diverse outputs

**ArgMax**: could be more stable in practice