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

Lecture 11: Word Vector Representations
Logistics

Assignment 3 ... was due last night
— This is the most difficult assignment in the course

Assignment 4 ... will be out today
— Do not wait

Assignment 5 ... will be delayed to enable project proposals
Logistics

**Paper readings** coming up

Project **groups** and topics

**Invited talks**
**Fun Example:** Code Deobfuscating with DOBF

**Obfuscated Code**

```python
class CLASS_0(nn.Module):
    def __init__(self, VAR_0, VAR_1, VAR_2, VAR_3):
        super(CLASS_0, self).__init__()
        VAR_0.VAR_1 = VAR_1
        VAR_0.VAR_2 = VAR_2
        VAR_0.VAR_4 = nn.Linear(VAR_1, (4 * VAR_2), bias=VAR_3)
        VAR_0.VAR_5 = nn.Linear(VAR_2, (4 * VAR_2), bias=VAR_3)
        VAR_0.FUNC_0()

def FUNC_0(VAR_6):
    VAR_7 = (1.0 / math.sqrt(VAR_6.VAR_8))
    for VAR_9 in VAR_6.VAR_10():
        VAR_9.data.uniform_((VAR_7), (VAR_7))

def FUNC_1(VAR_11, VAR_12, VAR_13):
    (VAR_14, VAR_15) = VAR_13
    VAR_14 = VAR_14.view(VAR_14.size(1), (1))
    VAR_15 = VAR_15.view(VAR_15.size(1), (1))
    VAR_12 = VAR_12.view(VAR_12.size(1), (1))
    VAR_16 = (VAR_11.VAR_12 + VAR_11.VAR_5(VAR_14))
    VAR_17 = VAR_16[:, (3 * VAR_11.VAR_8)].sigmoid()
    VAR_18 = VAR_16[:, (3 * VAR_11.VAR_8)].tanh()
    VAR_19 = VAR_17[:, VAR_11.VAR_8]
    VAR_20 = VAR_17[:, VAR_11.VAR_8](2 * VAR_11.VAR_8)
    VAR_21 = VAR_17[:, VAR_11.VAR_8]
    VAR_22 = (th.mul(VAR_15, VAR_20) + th.mul(VAR_19, VAR_18))
    VAR_23 = th.mul(VAR_21, VAR_22.tanh())
    VAR_24 = VAR_23.view(1, VAR_23.size(0), (- 1))
    VAR_25 = VAR_22.view(1, VAR_22.size(0), (- 1))
    return (VAR_23, (VAR_23, VAR_22))
```

**Code Deobfuscated using DOBF**

```python
class LSTM(nn.Module):
    def __init__(self, input_size, hidden_size, bias):
        super(LSTM, self).__init__()
        self.input_size = input_size
        self.hidden_size = hidden_size
        self.h1 = nn.Linear(input_size, (4 * hidden_size), bias=bias)
        self.h2 = nn.Linear(hidden_size, (4 * hidden_size), bias=bias)
        self.init_weights()

def init_weights(self):
    stdv = (1.0 / math.sqrt(self.hidden_size))
    for m in self.modules():
        m.data.uniform_((- stdv), stdv)

def forward(self, x, prev_state):
    prev_h = prev_h.view(prev_h.size(1), (- 1))
    prev_c = prev_c.view(prev_c.size(1), (- 1))
    x = x.view(x.size(1), (- 1))
    h = (self.h1(x) + self.h2(prev_h)).sigmoid()
    c = h[:, (3 * self.hidden_size)].tanh()
    r = (self.hidden_size)
    g = (self.hidden_size:2 + self.hidden_size)
    o = s[:, (- self.hidden_size)]
    c = (th.mul(prev_c, g) + th.mul(r, c))
    h = th.mul(o, c.tanh())
    return (h, (c, c))
```

[Roziere et al., ArXiv, 2021]
Representing a **Word**: One Hot Encoding

**Vocabulary**

dog
cat
person
holding
tree
computer
using

*slide from V. Ordonex*
Representing a **Word**: One Hot Encoding

**Vocabulary**

- dog: 1
- cat: 2
- person: 3
- holding: 4
- tree: 5
- computer: 6
- using: 7

*slide from V. Ordonex*
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

**bag-of-words** representation

<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*
Representing **Phrases**: Bag-of-Words

**bag-of-words** representation

person holding dog \( \{3, 4, 1\} \)  

\[ 1, 0, 1, 1, 0, 0, 0, 0, 0, 0 \]

---

**Vocabulary**

- dog 1
- cat 2
- person 3
- holding 4
- tree 5
- computer 6
- using 7

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

**bag-of-words** representation

<table>
<thead>
<tr>
<th>Phrase</th>
<th>Index Set</th>
<th>Representation</th>
</tr>
</thead>
<tbody>
<tr>
<td>person holding dog</td>
<td>{3, 4, 1}</td>
<td>[ 1, 0, 1, 1, 0, 0, 0, 0, 0 ]</td>
</tr>
<tr>
<td>person holding cat</td>
<td>{3, 4, 2}</td>
<td>[ 1, 1, 0, 1, 0, 0, 0, 0, 0 ]</td>
</tr>
</tbody>
</table>

**Vocabulary**

- dog: 1
- cat: 2
- person: 3
- holding: 4
- tree: 5
- computer: 6
- using: 7

*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>
</tbody>
</table>

### Bag-of-Words Representation

<table>
<thead>
<tr>
<th>dog</th>
<th>cat</th>
<th>person</th>
<th>holding</th>
<th>tree</th>
<th>computer</th>
<th>using</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

### Vocabulary

- dog: 1
- cat: 2
- person: 3
- holding: 4
- tree: 5
- computer: 6
- using: 7

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

<table>
<thead>
<tr>
<th>Phrase</th>
<th>Vocabulary</th>
<th>Bag-of-Words Representation</th>
</tr>
</thead>
<tbody>
<tr>
<td>person holding dog</td>
<td>dog, person, holding, tree</td>
<td>{3, 4, 1} [1, 0, 1, 1, 0, 0, 0, 0, 0]</td>
</tr>
<tr>
<td>person holding cat</td>
<td>cat, person, holding, computer</td>
<td>{3, 4, 2} [1, 1, 0, 1, 0, 0, 0, 0, 0]</td>
</tr>
<tr>
<td>person using computer</td>
<td>computer, using</td>
<td>{3, 7, 6} [0, 0, 0, 1, 0, 1, 1, 0, 0]</td>
</tr>
<tr>
<td>person using computer, person holding cat</td>
<td>dog, cat, person, holding, computer, using</td>
<td>{3, 3, 7, 6, 2} [0, 1, 2, 1, 0, 1, 1, 0, 0]</td>
</tr>
</tbody>
</table>

*slide from V. Ordonex*
Word Representations

1. **One-hot encodings** — only non-zero at the index of the word
   
   e.g., \([ 0, 1, 0, 0, 0, \ldots, 0, 0, 0 ]\)
   
   **Good:** simple
   
   **Bad:** not compact, distance between words always same (e.g., synonyms vs. antonyms)

2. **Word feature representations** — manually define “good” features
   
   e.g., \([ 1, 1, 0, 30, 0, \ldots, 0, 0, 0 ]\) -> 300-dimensional irrespective of dictionary
   
   e.g., word ends on -ing

3. **Learned word representations** — vector should approximate “meaning” of the word
   
   e.g., \([ 1, 1, 0, 30, 0, \ldots, 0, 0, 0 ]\) -> 300-dimensional irrespective of dictionary
   
   **Good:** compact, distance between words is semantic

* Adopted from slides by Louis-Philippe Morency
Distributional Hypothesis  [Lenci, 2008]

— 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.

* Adopted from slides by Louis-Philippe Morency
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.

* Adopted from slides by Louis-Philippe Morency
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 Euclidian space

Co-occurrence Matrix

* 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

* 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)
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 Euclidian space

Co-occurrence Matrix

* 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 Euclidian space

Co-occurrence Matrix

Way too high dimensional!
SVD for Dimensionality Reduction

$$X = U S V^T$$

$$\hat{X} = \hat{U} \hat{S} \hat{V}^T$$

*slide from Vagelis Hristidis*
Learned Word Vector Visualization

We can also use other methods, like LLE here:

Issues with SVD

**Computational** cost for a $d \times n$ matrix is $O(dn^2)$, 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

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

Key idea: Predict surrounding words of every word

Benefits: Faster and easier to incorporate new document, words, etc.
**word2vec**: Representing the Meaning of Words [Mikolov et al., 2013]

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

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

Continuous Bag of Words (**CBOW**): 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*
**CBOW: Continuous Bag of Words**

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

![Diagram](image)

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

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

- **Input layer**
  - Input vectors: `x ∈ R^|V|`
  - Word embeddings: `w_{|V|×|N|}`

- **Hidden layer**
  - Context vector: `v ∈ R^|N|`
  - Weight matrix: `w'_{|N|×|V|}`

- **Output layer**
  - Predicted labels: `y ∈ R^|V|`
  - Weight matrix: `w'_{|N|×|V|}`

*slide from Vagelis Hristidis

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

- **Input layer**
  - `x ∈ ℝ^|V|`
  - `W ∈ ℝ^|V|×|N|`

- **Hidden layer**
  - `v ∈ ℝ^|N|`
  - `W' ∈ ℝ^|N|×|V|`

- **Output layer**
  - `y ∈ ℝ^|V|`
  - `y' ∈ ℝ^|V|`

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

Input layer

Hidden layer

Parameters to be learned

Output layer

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

Parameters to be learned

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

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

[ Mikolov et al., 2013 ]

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

Input layer

\[
\begin{align*}
x_{\text{cat}} &= \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ \vdots \\ 0 \end{bmatrix} \\
x_{\text{on}} &= \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ \vdots \end{bmatrix}
\end{align*}
\]

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

\[
W_{|V|\times|N|}^T \times x_{\text{cat}} = v_{\text{cat}}
\]

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

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

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

- **Input layer**
  - $x_{cat}$
  - $x_{on}$

- **Hidden layer**
  - $v_{cat}$
  - $v_{on}$
  - $\hat{v} = \frac{v_{cat} + v_{on}}{2}$

- **Output layer**
  - $\hat{y} \in \mathbb{R}^{V}$
  - $y \in \mathbb{R}^{N}$

*slide from Vagelis Hristidis

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

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

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

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

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

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

\[ W'_{|V| \times |N|} \times \hat{v} = z \]

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

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

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

![Diagram](image)

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

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

Optimize to get close to 1-hot encoding

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

- Input layer
- Hidden layer
- Output layer

\[ x_{\text{cat}} \]
\[ x_{\text{on}} \]
\[ W^T_{|V| \times |N|} \]
\[ v_{\text{on}} = W^T_{|V| \times |N|} \times x_{\text{on}} \]
\[ \hat{v} \in \mathbb{R}^{|N|} \]
\[ v_{\text{on}} = W^T_{|V| \times |N|} \times x_{\text{on}} \]
\[ \hat{y} \in \mathbb{R}^{|V|} \]
\[ \hat{y} = \text{softmax}(z) \]
\[ \hat{y}_{\text{sat}} \]

*slide from Vagelis Hristidis

[ Mikolov et al., 2013 ]
CBOW: Interesting Observation

There are two representations for same word!

*slide from Vagelis Hristidis

[Mikolov et al., 2013]
Another way to look at it: Maximize similarity between context word representation and the word representation itself.

\[
p(w|c) = \frac{\exp \left[ \sum_c (Wx_c)^T (Wx_w) \right]}{\sum_{i=1}^{V} \exp \left[ (Wx_i)^T (Wx_w) \right]}
\]
CBOB: Interesting Observation

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

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

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

[ Mikolov et al., 2013 ]
Comparison

- **CBOW** is not great for rare words and typically needs less data to train.
- **Skip-gram** better for rare 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>Collobert-Weston NNL</td>
<td>50</td>
<td>660M</td>
<td>9.3</td>
</tr>
<tr>
<td>Turian NNL</td>
<td>50</td>
<td>37M</td>
<td>1.4</td>
</tr>
<tr>
<td>Turian NNL</td>
<td>200</td>
<td>37M</td>
<td>1.4</td>
</tr>
<tr>
<td>Mnih NNL</td>
<td>50</td>
<td>37M</td>
<td>1.8</td>
</tr>
<tr>
<td>Mnih NNL</td>
<td>100</td>
<td>37M</td>
<td>3.3</td>
</tr>
<tr>
<td>Mikolov RNNLM</td>
<td>80</td>
<td>320M</td>
<td>4.9</td>
</tr>
<tr>
<td>Mikolov RNNLM</td>
<td>640</td>
<td>320M</td>
<td>8.6</td>
</tr>
<tr>
<td>Huang NNL</td>
<td>50</td>
<td>990M</td>
<td>13.3</td>
</tr>
<tr>
<td>Our NNL</td>
<td>20</td>
<td>6B</td>
<td>12.9</td>
</tr>
<tr>
<td>Our NNL</td>
<td>50</td>
<td>6B</td>
<td>27.9</td>
</tr>
<tr>
<td>Our NNL</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><strong>50.0</strong></td>
</tr>
</tbody>
</table>
Interesting Results: **Word Analogies**

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

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

\[ a:b :: c:? \]

- man:woman :: king:?
  - king \[ [0.30 0.70] \]
  - man \[ [0.20 0.20] \]
  - woman \[ [0.60 0.30] \]
  - queen \[ [0.70 0.80] \]
Interesting Results: **Word Analogies**

[ Mikolov et al., 2013 ]
Dynamic Word Embeddings

190

- broadcast
- scattered
- burnt
- dispersed
- burned
- distributed
Dynamic Word Embeddings

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

Lecture 11: RNN Applications
Let us look at some actual practical uses of RNNs
Applications: Skip-thought Vectors

word2vec but for sentences, where each sentence is processed by an LSTM

[Kiros et al., 2015]
Applications: Google Language Translation

One model to translate from **any language** to any other language

[Johnson et al., 2017]
Applications: Google Language Translation

One model to translate from **any language** to any other language

[Johnson et al., 2017]
Applications: Google Language Translation

One model to translate from any language to any other language

Flipped order encoding
Token designating target language

[Johnson et al., 2017]
Applications: Google Language Translation

One model to translate from **any language** to any other language

- Flipped order encoding
- Token designating target language

[Johnson et al., 2017]
Applications: Google Language Translation

One model to translate from **any language** to any other language

Flipped order encoding

Token designating **target** language

8! layer LSTM decoder and encoder

[Johnson et al., 2017]
Applications: Google Language Translation

One model to translate from **any language** to any other language

Residual at other layers (ResNet style)

Bi-directional at lower layers

Flipped order encoding

Token designating **target** language

8! layer LSTM decoder and encoder

[ Johnson et al., 2017 ]
Applications: Google Language Translation

One model to translate from **any language** to any other language

- **Residual** at other layers (ResNet style)
- **Bi-directional** at lower layers
- **Flipped** order encoding

Token designating **target** language

8! layer LSTM decoder and encoder

[Johnson et al., 2017]
Applications: BERT and SoTA

To learn relationships between sentences, predict whether Sentence B is actual sentence that **proceeds** Sentence A, or a random sentence

Sentence A = The man went to the store.
Sentence B = He bought a gallon of milk.
Label = IsNextSentence

Sentence A = The man went to the store.
Sentence B = Penguins are flightless
Label = NotNextSentence
Applications: BERT and SoTA

Use 30,000 WordPiece vocabulary

Each token is a sum of three embeddings
Applications: BERT and SoTA

Multi-headed self **attention**
- Models context

Feed-forward layers
- Computes non-linear hierarchical features

Layer norm and **residuals**
- Makes training deep neural network (e.g., 12 layers possible)

**Positional Embeddings**
- Allows model to learn relative positioning
Applications: BERT and SoTA

Pre-training
Applications: BERT and SoTA
Applications: BERT and SoTA
Applications: BERT and SoTA

Transfer: ASNQ Dataset

Adapt: Target Dataset
Applications: Neural Image Captioning

* slide from Dhruv Batra
Applications: Neural Image Captioning

Image Embedding (VGGNet)
Applications: Neural Image Captioning

Image Embedding (VGGNet)
Applications: Neural Image Captioning

* slide from Dhruv Batra
Applications: Neural Image Captioning

Two people and two horses.

* slide from Dhruv Batra
Applications: Neural Image Captioning

Good results

A cat sitting on a suitcase on the floor
A cat is sitting on a tree branch
A dog is running in the grass with a frisbee
A white teddy bear sitting in the grass

Two people walking on the beach with surfboards
A tennis player in action on the court
Two giraffes standing in a grassy field
A man riding a dirt bike on a dirt track

* slide from Fei-Dei Li, Justin Johnson, Serena Yeung, cs231n Stanford
Applications: Neural Image Captioning

Failure cases

A woman is holding a cat in her hand

A person holding a computer mouse on a desk

A woman standing on a beach holding a surfboard

A bird is perched on a tree branch

A man in a baseball uniform throwing a ball

* slide from Fei-Dei Li, Justin Johnson, Serena Yeung, cs231n Stanford
Applications: Image Captioning with Attention

RNN focuses its attention at a different spatial location when generating each word

[ Xu et al., ICML 2015 ]
Applications: Image Captioning with Attention

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

[ Xu et al., ICML 2015 ]
Applications: Image Captioning with Attention

* slide from Fei-Dei Li, Justin Johnson, Serena Yeung, cs231n Stanford
Applications: Image Captioning with Attention

\[ z = \sum_{i=1}^{L} p_i v_i \]
Applications: Image Captioning with Attention

* slide from Fei-Dei Li, Justin Johnson, Serena Yeung, cs231n Stanford
Applications: Image Captioning with Attention

[Xu et al., ICML 2015]
Applications: Image Captioning with Attention

[ Xu et al., ICML 2015 ]
**Applications:** Image Captioning with Attention

Image Captioning with Attention

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

[ Xu et al., ICML 2015 ]

Image: H x W x 3

CNN

Distribution over L locations

Distribution over vocab

a1

a2

d1

a3

d2

h0

h1

h2

\cdots

Features: L x D

Weighted features: D

Weighted combination of features

First word

z1

y1

z2

y2
Applications: Image Captioning with Attention

[ Xu et al., ICML 2015 ]
Applications: Image Captioning with Attention

Good results

A woman is throwing a frisbee in a park.

A dog is standing on a hardwood floor.

A stop sign is on a road with a mountain in the background.

A little girl sitting on a bed with a teddy bear.

A group of people sitting on a boat in the water.

A giraffe standing in a forest with trees in the background.

[Xu et al., ICML 2015]
Applications: Image Captioning with Attention

Failure results

A large white bird standing in a forest.

A woman holding a clock in her hand.

A man wearing a hat and a hat on a skateboard.

A person is standing on a beach with a surfboard.

A woman is sitting at a table with a large pizza.

A man is talking on his cell phone while another man watches.

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