Unstructured Data Analysis

Recitation: Sentiment analysis with IMDb reviews; word embeddings; a look at some PyTorch code

George Chen
Recitation

• Demo: sentiment analysis with IMDb reviews

• More on word embeddings & fine tuning

• (Time permitting) A little bit of what’s under the hood: UDA_pytorch_utils.py
(From Lecture) Sentiment Analysis with IMDb Reviews

Step 1: Tokenize & build vocabulary

<table>
<thead>
<tr>
<th>Word index</th>
<th>Word</th>
<th>2D Embedding</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>this</td>
<td>[-0.57, 0.44]</td>
</tr>
<tr>
<td>1</td>
<td>movie</td>
<td>[0.38, 0.15]</td>
</tr>
<tr>
<td>2</td>
<td>rocks</td>
<td>[-0.85, 0.70]</td>
</tr>
<tr>
<td>3</td>
<td>sucks</td>
<td>[-0.26, 0.66]</td>
</tr>
</tbody>
</table>

Step 2: Encode each review as a sequence of word indices into the vocab

“this movie sucks” ➞ 0 1 3

Step 3: Use word embeddings to represent each word

[-0.57, 0.44]
[0.38, 0.15]
[-0.26, 0.66]
(From Lecture) Sentiment Analysis with IMDb Reviews

Step 1: Tokenize & build vocabulary

<table>
<thead>
<tr>
<th>Word index</th>
<th>Word</th>
<th>2D Embedding</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>this</td>
<td>[-0.57, 0.44]</td>
</tr>
<tr>
<td>1</td>
<td>movie</td>
<td>[0.38, 0.15]</td>
</tr>
<tr>
<td>2</td>
<td>rocks</td>
<td>[-0.85, 0.70]</td>
</tr>
<tr>
<td>3</td>
<td>sucks</td>
<td>[-0.26, 0.66]</td>
</tr>
</tbody>
</table>

In the demo, this part done by creating an instance of the SpacyEncoder Python class (torchtext does support other encoders as well in case you don’t like spacy/spacy is giving you trouble)

Step 2: Encode each review as a sequence of word indices into the vocab

“this movie sucks” → 0 1 3

Step 3: Use word embeddings to represent each word

embedding_weights (100-dimensional GloVe embeddings in the demo)

[-0.57, 0.44]
[0.38, 0.15]
[-0.85, 0.70]
[-0.26, 0.66]
Sentiment Analysis with IMDb Reviews

Demo
Word Embeddings:
Even without labels, we can set up a prediction problem!

Hide part of training data and try to predict what you’ve hid!
Word Embeddings: word2vec

Can solve tasks like the following:

Man is to King as Woman is to ______
Word Embeddings: word2vec

Can solve tasks like the following:

Man is to King as Woman is to Queen
Word Embeddings: word2vec

Can solve tasks like the following:

Man is to King as Woman is to Queen

Which word doesn’t belong?
blue, red, green, crimson, transparent
Word Embeddings: word2vec

Can solve tasks like the following:

Man is to King as Woman is to Queen

Which word doesn’t belong? blue, red, green, crimson, transparent
Word Embeddings: word2vec

Image source: https://deeplearning4j.org/img/countries_capitals.png
The opioid epidemic or opioid crisis is the rapid increase in the use of prescription and non-prescription opioid drugs in the United States and Canada in the 2010s.

Predict context of each word!

Training data point: epidemic

“Training labels”: the, opioid, or, opioid
Word Embeddings: word2vec

The opioid epidemic, or opioid crisis, is the rapid increase in the use of prescription and non-prescription opioid drugs in the United States and Canada in the 2010s.

Predict context of each word!

Training data point: or

“Training labels”: opioid, epidemic, opioid, crisis
The opioid epidemic or opioid crisis is the rapid increase in the use of prescription and non-prescription opioid drugs in the United States and Canada in the 2010s.

Predict context of each word!

Training data point: opioid

“Training labels”: epidemic, or, crisis, is

There are “positive” examples of what context words are for “opioid”

Also provide “negative” examples of words that are not likely to be context words (by randomly sampling words elsewhere in document)
The opioid epidemic or opioid crisis is the rapid increase in the use of prescription and non-prescription opioid drugs in the United States and Canada in the 2010s.

Predict context of each word!

Training data point: opioid

“Negative training label”: 2010s

Also provide “negative” examples of words that are not likely to be context words (by randomly sampling words elsewhere in document).
Use one-hot encoding:

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

vector length = vocab size

index of "opioid" in vocab

Linear (100 nodes)

Learned weight matrix used as word embedding!

(Treat i-th row of weight matrix as word embedding for i-th word)

Linear (vocab size)

Want real context words (e.g., "epidemic", "crisis") to have high probability
Word Embeddings as a Special Case of Self-Supervised Learning

- Key idea: hide part of the training data and try to predict hidden part using other parts of the training data.

- No actual training labels required — we are defining what the training labels are just using the unlabeled training data!

- This is an *unsupervised* method that sets up a *supervised prediction* task.

- Other word embeddings methods are possible.
What about a word that has multiple meanings?

Challenging: try to split up word into multiple words depending on meaning (requires inferring meaning from context)

This problem is called word sense disambiguation (WSD)
Word Embeddings as a Special Case of Self-Supervised Learning

- Key idea: hide part of the training data and try to predict hidden part using other parts of the training data
- No actual training labels required — we are defining what the training labels are just using the unlabeled training data!
- This is an *unsupervised* method that sets up a *supervised prediction* task
- Other word embeddings methods are possible
  - Word embedding that handles word-sense disambiguation: BERT (current state of the art)
  - **Warning:** the default PyTorch Embedding layer does *not* do anything clever like BERT/GloVe/word2vec (best to use pre-trained word embeddings!)
(From Lecture) Fine Tuning

Sentiment analysis RNN demo

Text → Embedding → Classifier → Positive/negative sentiment

We fixed the weights here to come from pre-trained GloVe word embeddings

GloVe vectors pre-trained on massive dataset (Wikipedia + Gigaword)
IMDb review dataset is small in comparison
(Flashback) Word2vec Neural Net

“opioid”
Use one-hot encoding
[0, 0, ..., 1, ..., 0]
vector length = vocab size
index of “opioid” in vocab

Linear
(100 nodes)
Learned weight matrix
used as word embedding!

Linear
(# nodes = vocab size),
Softmax

Want real context words (e.g.,
“epidemic”, “crisis”) to have high probability

(Treat i-th row of weight matrix as word embedding for i-th word)
(Flashback) Word2vec Neural Net

Turn off training

"opioid"
Use one-hot encoding
[0, 0, ..., 1, ..., 0]
vector length = vocab size
index of "opioid" in vocab

Linear
(100 nodes)
Learned weight matrix
used as word embedding!
(Treat i-th row of weight matrix as word embedding for i-th word)

Remove final layers
(Flashback) Word2vec Neural Net

**Turn off training**

```
“opioid”
Use one-hot encoding
[0, 0, …, 1, …, 0]
```

vector length = vocab size

index of “opioid” in vocab

```
Linear (100 nodes)
```

Learned weight matrix used as word embedding!

(Treat i-th row of weight matrix as word embedding for i-th word)

```
LSTM
Classifier
```

Add layers for new task we care about (such as sentiment analysis)
A Look Under the Hood

UDA_pytorch_utils.py