RNN vs CNN: SENTIMENT ANALYSIS

Restaurant Reviews

Candela García Quintero
Esther Flores Gonzalez
María Hernández Padilla

Idea

Build machine learning models to detect sentiment in review comments, and predict the quality of the product based on other reviews.

Two different models: RNN and CNN.

INDEX

Dataset

The fries were great too.	1
A great touch.	1
Service was very prompt.	1

Service stinks here!	0
waited and waited.	0
This place is not quality sushi, it is not a quality restaurant.	0

Customer Restaurant Reviews

Quality of food, service, place...

- CSV file: 1000 samples
- 2 columns:
  - 1. Customer Reviews
  - 2. Like:
    - 1 if positive
    - 0 if negative

Divide dataset into:

- Train
- Validate
- Test

Preparation of Dataset

TorchText: package consisting of data processing utilities for natural language. (NLP)

- **Fields**
  Preprocess text data types that can be converted to Tensor.
  - TEXT: processing of reviews
    Strings splitted into discrete tokens.
  - LABEL: processing of the sentiment

- **Vocabulary:** max of 25000 words
  Every unique word ⇒ index⇒ one-hot vector

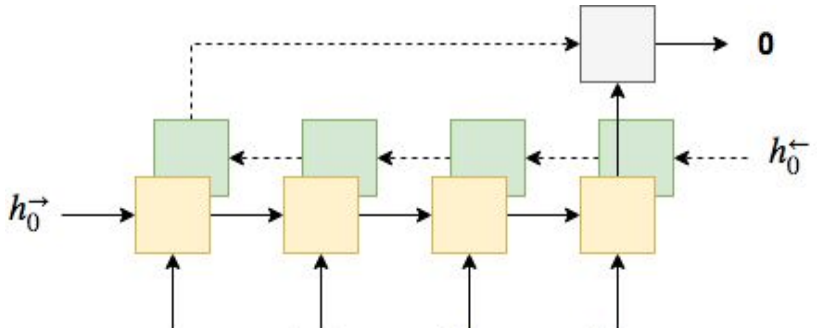
word	index	one-hot vector
I	0	[1, 0, 0, 0]
hate	1	[0, 1, 0, 0]
this	2	[0, 0, 1, 0]
restaurant	3	[0, 0, 0, 1]

Implementation of RNN model

LSTM (Long Short-Term Memory): does not suffer from vanishing gradient problem.

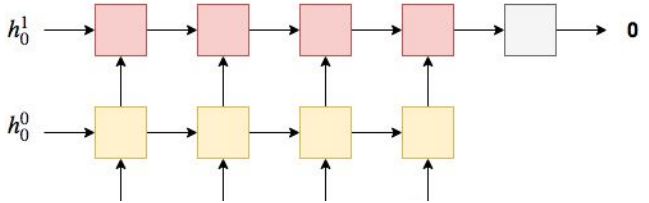
- **Bidirectional RNN:**
  - Forward RNN: One RNN processing words from the first to the last.
  - Backward RNN: processing words from the last to the first.
    Hidden state tensors returned by both RNNs create a single tensor.

Our sentiment prediction is made concatenating the last hidden state from the forward (final word of the sentence) with the last hidden state of the backward RNN (first word of the sentence).



Implementation of RNN model

LSTM (Long Short-Term Memory).



- **Multi-layer RNN:** we add additional RNNs on top of the initial RNN, where each RNN is another layer. The sentiment prediction is made from the final hidden state of the final layer.

  Our RNN model has 2 layers. (2 LSTMs stacked together)

- **Dropout:** regularization method to combat overfitting.
  - Randomly dropping out neurons in a layer during a forward pass.
  - The model becomes “weaker”, not over-parameterized.

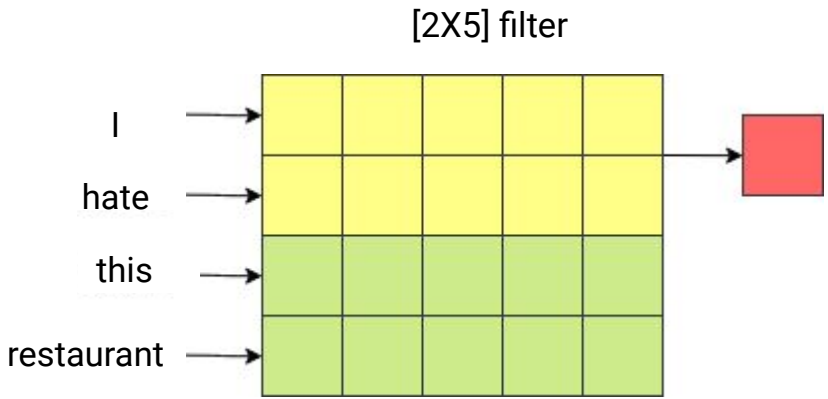
  Our RNN model implements a hyperparameter of probability of 0.5 of being dropped out.

Implementation of CNN model

LAYERS

- **Embedding:** phrases from the vocabulary are mapped to vectors of real numbers
  - transform one-hot vectors into a dense embedding vector.
    Reduces dimensionality of the input and, also words which have similar impact on the sentiment are mapped closer.
- **Convolutional Layer:**

Layers	Kernel Size [n x emb_dim]
1 Conv	3x100 (tri-grams)
2 Conv	4x100 (4-grams)
3 Conv	5x100 (5-grams)



Implementation of CNN model

- **Pooling layer (MaxPool) and ReLu activation:**
  The addition of a pooling layer after the convolutional layer is a common pattern used for ordering layers within a convolutional neural network that may be repeated one or more times in a given model.

  Applying the ReLU *after* the max-pooling reduces the number of operations.

- **Dropout layer**
  At each training stage, individual nodes are either dropped out of the net with probability 1-0’5 or kept with probability 0’5. Important not to use it in the last layer.

- **Linear Layer:**
  Weights up the evidence obtained and makes the final decision.

Results

For RNN

Accuracy of the test : 73.05%

```
[27] predict_sentiment(model, "I probably won't be coming back here.")

Negative review
0.15024596452713013
```

```
[28] predict_sentiment(model, "The chips and sals a here is amazing!!!!!!!!!!!!!!!!!!")

Positive review
0.9350672662258148
```

```
[29] predict_sentiment(model, "I hated it")

Negative review
0.34199678897857666
```

```
[30] predict_sentiment(model, "It sucks")

Positive review
0.5417261719703674
```

For CNN

Accuracy of the test : 77.73%

```
[65] predict_sentiment(model_cnn, "I probably won't be coming back here.")

Negative review
0.09052562713623047
```

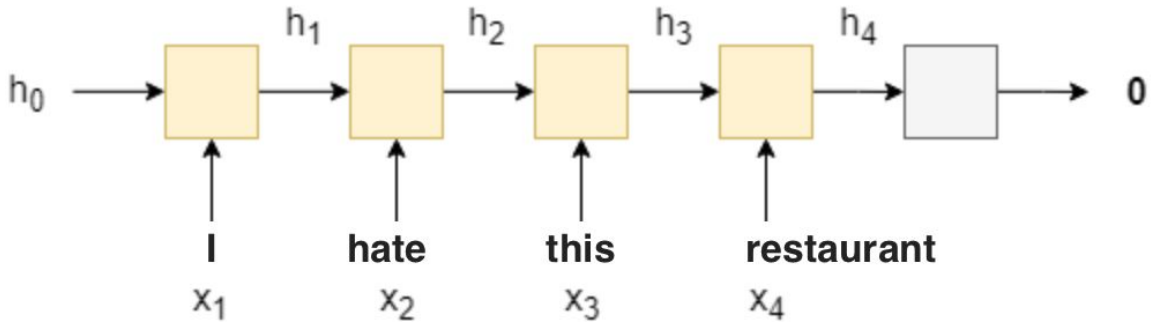
```
[66] predict_sentiment(model_cnn, "The chips and sals a here is amazing!!!!!!!!!!!!!!!!!")

Positive review
0.9720785040408373
```

```
[68] predict_sentiment(model_cnn, "I hated it")

Negative review
0.42455071210861206
```

```
predict_sentiment(model_cnn, "It sucks")

Negative review
0.38974159955978394
```

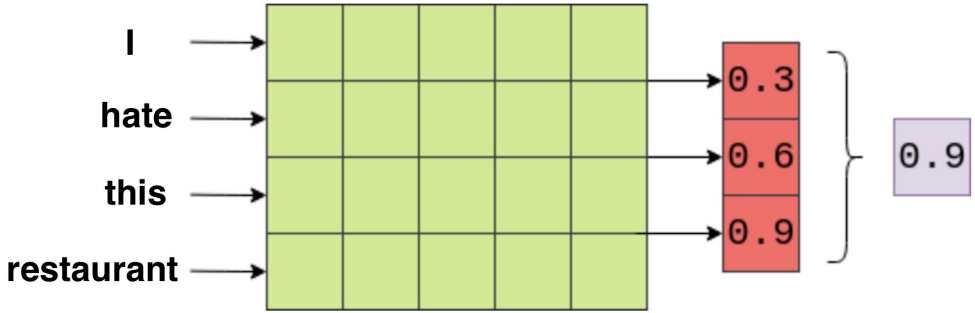
Why these differences?

An RNN takes in sequence of words, one at a time, and produces a hidden state, for each word. Once we have our final hidden state, we feed it through a linear layer, to receive our predicted sentiment.



Why these differences?

In here the appearance of certain bigrams (a 1x2 filter), tri-grams (a 1x3 filter) and n-grams (a 1x filter) within the review will be a good indication of the final sentiment.



The idea here is that the maximum value is the "most important" feature for determining the sentiment of the review, which corresponds to the "most important" n-gram within the review.

Why these differences?

CNN	RNN
It is suitable for spatial data such as images.	RNN is suitable for temporal data, also called sequential data.
CNN is considered to be more powerful than RNN.	RNN includes less feature compatibility when compared to CNN.
This network takes fixed size inputs and generates fixed size outputs.	RNN can handle arbitrary input/output lengths.
CNN is a type of feed-forward artificial neural network with variations of multilayer perceptrons designed to use minimal amounts of preprocessing.	RNN unlike feed forward neural networks - can use their internal memory to process arbitrary sequences of inputs.
CNNs use connectivity pattern between the neurons. This is inspired by the organization of the animal visual cortex, whose individual neurons are arranged in such a way that they respond to overlapping regions tiling the visual field.	Recurrent neural networks use time-series information - what a user spoke last will impact what he/she will speak next.
CNNs are ideal for images and video processing.	RNNs are ideal for text and speech analysis.

Why the accuracy is a bit low?

Data size is a very crucial part of NN. Larger datasets can help us better learn model parameters and improve the optimization process.

References

- Restaurant Customer Reviews, September 2019: https://www.kaggle.com/vigneshwarsofficial/reviews

- Updated Sentiment Analysis, Ben Trevett, César de Pablo, 19 Sep 2019, https://github.com/bentrevett/pytorch-sentiment-analysis/blob/master/2%20-%20Upgraded%20Sentiment%20Analysis.ipynb

- Convolutional Sentiment Analysis, Ben Trevett, 6 Nov 2019, https://github.com/bentrevett/pytorch-sentiment-analysis/blob/master/4%20-%20Convolutional%20Sentiment%20Analysis.ipynb

- https://pytorch.org/text/