A Practical Guide to Modern Natural Language Processing

Cristian Schuszter
Outline

1. Ancient history
2. Common tasks
3. Classical models & applications
4. Feature engineering
5. The deep learning years
6. State of the art
7. Q & A
The imitation game
ChatGPT broke the Turing test – the race is on for new ways to assess AI

Large language models mimic human chatter, but scientists disagree on their ability to reason.

By Celeste Slavov
What’s going on behind the scenes
The dark ages - rule-based systems

Welcome to

Eliza is a mock Rogerian psychotherapist. The original program was described by Joseph Weizenbaum in This implementation by Norbert Landsteiner 2005.

ELIZA: Is something troubling you?
YOU: Men are all alike.
ELIZA: What is the connection, do you suppose?
YOU: They're always bugging us about something or other.
ELIZA: Can you think of a specific example?
YOU: Well, my boyfriend made me come here.
ELIZA: Is it important to you that your boyfriend made you come here?
YOU: He says I'm depressed much of the time.
ELIZA: I am sorry to hear that you are depressed.
YOU: It's true. I am unhappy.
ELIZA: Can you explain what made you unhappy?
YOU:

https://github.com/oren/eliza-bot
The dark ages - rule-based systems (2)

https://patrickvanbergen.com/blocks-world/
Outline

1. Ancient history
2. Common tasks
3. Classical models & applications
4. Feature engineering
5. The deep learning years
6. State of the art
7. Q & A
Before the DL “revolution”

Natural Language Processing

Analysis Tasks
- Semantic
  - Named Entity Recognition
  - Similarity/relatedness
  - Text classification
  - Topic Modelling
  - Opinion/Sentiment analysis
- Syntactic Parsing
  - Part of Speech tagging
  - Chunking
  - Dependency parsing

Generation Tasks
- Question Answering
  - Chatbots
- Language Generation
  - Text generation
  - New words prediction
- Machine translation

After
Outline

1. Ancient history
2. Common tasks
3. **Classical models & applications**
4. Feature engineering
5. The deep learning years
6. State of the art
7. Q & A
Text classification - Naive Bayes

\[ P(A|B) = \frac{P(B|A) \cdot P(A)}{P(B)} \]

- The probability of "B" being true given that "A" is true
- The probability of "A" being true
- The probability of "B" being true

Email

Machine Learning Model

Spam

Not Spam
Naive Bayes

Naive = words in the sentence are independent

Spam example:
("Hello, this is a normal email", 0),
("Congratulations! You've won a free vacation", 1),
("Click here to claim your prize", 1),
("Meeting scheduled for tomorrow", 0),
("Limited time offer, buy now", 1)

\[ P(\text{spam}) = \frac{3}{5} \]
\[ P(\text{ham}) = \frac{2}{5} \]

\[ P(\text{SPAM} \mid w) = \frac{P(w \mid \text{SPAM}) \cdot P(\text{SPAM})}{P(w \mid \text{SPAM}) \cdot P(\text{SPAM}) + P(w \mid \text{HAM}) \cdot P(\text{HAM})} \]
Outline

1. Ancient history
2. Common tasks
3. Classical models & applications
4. Feature engineering
5. The deep learning years
6. State of the art
7. Q & A
Feature engineering

- Highly essential for building useful ML algorithms
- Range from simple to probabilistic distributions
Vector space models for text

The formula for cosine similarity

\[
similarity = \cos(\theta) = \frac{A \cdot B}{\|A\|_2 \|B\|_2} = \frac{\sum_{i=1}^{n} A_i B_i}{\sqrt{\sum_{i=1}^{n} A_i^2} \sqrt{\sum_{i=1}^{n} B_i^2}}
\]

© Cristian Schuszter & CERN, ICSC, 15.04.2024
One-hot encoding

- **the** [1,0,0,0]
- **dog** [0,1,0,0]
- **is** [0,0,1,0]
- **grey** [0,0,0,1]

- 
  ```plaintext
  [1, 0, 0, 0],
  [0, 1, 0, 0],
  [0, 0, 1, 0],
  [0, 0, 0, 1]
  ```
Bag of words

dog and dog are friends

dog x 2

and x 1

are x 1

friends x 1

[2,1,1,1]
Bag of n-grams

2-grams are called “bigrams”

To be, or not to be, that is the question

{("To","be"),
("be","or"),
("or","not"),
("not","to"),
("to","be"),
("be","that"),
("that","is"),
("is","the"),
("the","question"))
TF-IDF

Term Frequency - Inverse Document Frequency

Example:

Our document contains 100 words, 3 times cat.

\[ TF(cat) = \frac{3}{100} = 0.03 \]

The database contains 10 million documents, and cat is in 1000 of those.

\[ IDF(cat) = \log_{10}(10,000,000 / 1000) = 4 \]

\[ TF_{IDF}(cat) = TF(cat) \times IDF(cat) = 4 \times 0.03 = 0.12 \]
What can I do with this?

- **Topic modelling**
  - Compute TF-IDF vectors of the documents in your corpus
  - Use a dimensionality reduction technique to visualize them

- **Simple text classification**

- **Simple search engines**
  - Pre-compute tf-idf for all your documents
  - Query comes in -> compute TF-IDF for the query
  - Find top k documents

![Diagram showing topic modeling results](Image)
Outline

1. Ancient history
2. Common tasks
3. Classical models & applications
4. Feature engineering
5. The deep learning years
6. State of the art
7. Q & A
But how can we capture meaning?

Feature engineering 2.0 - using deep learning

![Diagram showing feature engineering using deep learning](https://via.placeholder.com/150)
Word2Vec

Intuition: Given a document, for each word represent

- The probability of obtaining a word given the context around it (CBOW)
  - The sum of all the context should give this word as the most likely
- The probability of obtaining the context words given a word (Skip-gram)
  - More computationally expensive, better results in training
- Back-propagate (train the model) and adjust the representation of the word
Word2Vec: CBOW
Word2Vec: Skip-gram

outside context words in window of size 2

outside context words in window of size 2

center word at position t
Similar algorithms

- **GloVe: Global Vectors for Word Representation**
  - Using global word-to-word co-occurrence in the corpus for training
  - Gradient descent for training

- **Fasttext**
  - Solves the problem of OOV (out of vocabulary) words.
  - Uses sub-words in the training and embedding process to make “educated guesses”

<table>
<thead>
<tr>
<th>English</th>
<th>anarchy</th>
<th>chy</th>
<th>&lt;anar</th>
<th>narchy</th>
</tr>
</thead>
<tbody>
<tr>
<td>monarchy</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>kindness</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>politeness</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>unlucky</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>lifetime</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>starfish</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>submarine</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>transform</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>EN</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>&lt;unlucky</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>&lt;life</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>&lt;fish</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>&lt;sub</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>&lt;form</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Visualizing your word embeddings

**Problem:** High-dimensional data (300+ dimensions)

How do we get over this? **Dimensionality reduction techniques**

Most popular: Principal Component Analysis (PCA) & tSNE

https://distill.pub/2016/misread-tsne/
https://projector.tensorflow.org/
Deep learning models

- Mainly 2 use-cases:
  - Classification
  - Language modeling (question answering, translation)
- Much more performant than classical techniques

Caveats

- Computationally expensive to train
  - Many parameters for the weights
  - Hard to parallelize
- Doesn’t do well on long inputs

Example classification architecture
Text CNN

https://arxiv.org/abs/1408.5882
Encoder - decoder models - RNN, GRU, LSTM

I am a student

Je suis étudiant
GRU vs LSTM cells

GRU cell

\[ W_{ax} X(t) \]

\[ X(t) \]

\[ a(t-1) \]

\[ a(t) \]

\[ ^\wedge a(t) \]

\[ y(t) \]

update gate

softmax

tanh

LSTM cell

\[ W_{ax} X(t) \]

\[ X(t) \]

\[ c(t-1) \]

\[ c(t) \]

\[ a(t-1) \]

\[ a(t) \]

\[ ^\wedge a(t) \]

\[ y(t) \]

softmax

forget g.

update g.

tanh

output g.
Outline

1. Ancient history
2. Common tasks
3. Classical models & applications
4. Feature engineering
5. The deep learning years
6. State of the art
7. Q & A
Transformers

- Recurrent / convolutional models are really slow to train and don’t scale

- Intuitive example: translation
  - Language A gets encoded
  - Gets decoded into sentence for Language B

https://arxiv.org/abs/1706.03762
Attention & masked language model training

![Diagram of masked language model training](image)
Thank you!

Contact:

- Emails:
  - Work: cristian.schuszter@cern.ch
  - Not work: chrisschuszter@gmail.com

© Cristian Schuszter & CERN, iCSC, 15.04.2024